Universidad Zaragoza

Trabajo Fin de Máster

Desarrollo de un microservicio para la caracterización de lesiones de psoriasis y su integración sobre la red social privada SIGNAL.

Development of a microservice for the characterization of psoriasis lesions and its integration on the private social network SIGNAL.

Autor/es

Daniel Vicente Moya

Director/es

José García Moros
Javier Martínez Torres

Escuela de Ingeniería y Arquitectura / Universidad de Zaragoza

2018

Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

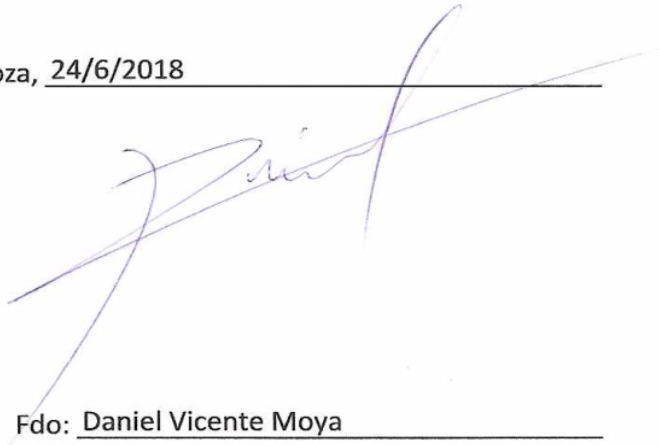
D./Dª. Daniel Vicente Moya,

con nº de DNI 18451494-C en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Ingeniería Biomédica, (Título del Trabajo) Desarrollo de un microservicio para la caracterización de lesiones de psoriasis y su integración sobre la red social privada SIGNAL.

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 24/6/2018



Fdo: Daniel Vicente Moya

Desarrollo de un microservicio para la caracterización de lesiones de psoriasis y su integración sobre la red social privada SIGNAL.

Resumen

Tras la última revolución en la comunicación, se ha extendido ampliamente a nivel global el empleo de plataformas de comunicación, como WhatsApp o Telegram. Gracias a esto se ha facilitado la forma de comunicarse a larga distancia, abriendo nuevas posibilidades de aplicación que llegan incluso al ámbito médico de los sistemas e-Health.

Este Trabajo de Fin de Máster continúa los recientes trabajos que emplean el uso de plataformas de mensajería, en este caso la plataforma Signal, como medio para la monitorización de pacientes en el ámbito médico. Se plantea el uso de *bots* (programas informáticos que ejecutan tareas automatizadas mediante la interacción con el usuario) que ofrecen al usuario servicios y funcionalidades para su propia autogestión en el seguimiento de la enfermedad. En concreto, este trabajo se centra en el desarrollo de una funcionalidad que ofrecerá al usuario la opción de realizar un seguimiento de las lesiones de psoriasis. Esto se consigue gracias al desarrollo de la plataforma global en una arquitectura software de microservicios con una comunicación mediante el protocolo HTTP, que permite integrar servicios como el aquí desarrollado para añadir la funcionalidad deseada.

Mediante la interacción con el *bot*, el usuario seleccionará la funcionalidad y enviará una imagen para su análisis. La imagen será tratada mediante procesos de segmentación y binarizado, que analizaran las características de la psoriasis, permitiendo extraer una máscara para la detección de las lesiones y extrayendo la información relativa a la evolución de la extensión de la lesión en la superficie de la piel. Este análisis vendrá determinado por la información facilitada por el usuario en referencia al tipo de psoriasis que padece. Se ha planteado que el análisis realizado en el trabajo se centre en las formas más comunes de psoriasis. Esto establece dos análisis según la tipología de lesión: análisis tipo gota, que se centra más en las regiones enrojecidas de la piel y análisis tipo placa, que se centra en detectar las zonas blancas formadas por la placa además de las zonas de piel enrojecida.

Con el objetivo de demostrar la viabilidad del microservicio en el ámbito de la monitorización, se desarrollan pruebas de concepto de los análisis que ofrece la funcionalidad introducida en la plataforma.

Development of a microservice for the characterization of psoriasis lesions and its integration on the private social network SIGNAL.

Summary

After the last revolution in communication, the use of communication platforms, such as WhatsApp or Telegram, has spread widely on a global level. Thanks to this, the way of communicating at long distance, opening new application possibilities that even reach the medical field of e-Health systems.

This Master's Thesis continues the recent works that uses messaging platforms, in this case the Signal platform, as a means of monitoring patients in the medical field. The use of bots (computer programs that perform automated tasks through interaction with the user) is proposed that offer the user services and functionalities for their own self-management in the disease. In particular, this work focuses on the development of a functionality that will offer the user the option to track psoriasis lesions. This is achieved thanks to the development of the global platform in a software architecture of microservices with a communication through the HTTP protocol, that allows to integrate services such as the one developed here to add the desired functionality .

Through interaction with the bot, the user will select the functionality and send the image for its analysis. The image will be treated by means of segmentation and binarization processes, which will analyze the characteristics of psoriasis, allowing the extraction of a mask for the detection of lesions and extracting information regarding the evolution of the extension of the lesion on the surface of the skin. This analysis will be determined by the information provided by the user in reference to the type of psoriasis suffered. It is proposed that the analysis carried out in the work focus on the most common forms of psoriasis. This establishes two analyzes according to the type of injury: drop type analysis, which focuses more on the reddened regions of the skin and plaque type analysis, which focuses on detecting the white areas formed by the plaque in addition to the areas of reddened skin.

In order to demonstrate the viability of the microservice in the field of monitoring, concept tests of the analysis offered by the functionality introduced in the platform are developed.

Tabla de contenido

Índice de figuras

Índice de tablas

Índice de códigos

1. Introducción

Hoy en día las nuevas tecnologías de comunicación han permitido crear una generación de dispositivos móviles con los que somos capaces de realizar desde la más básica de las llamadas, a realizar fotografías y a ejecutar aplicaciones instaladas en el dispositivo. Son estas aplicaciones, en especial las de mensajería, las que permiten ampliar las funcionalidades de estos dispositivos móviles llegando a alcanzar el ámbito de la medicina. Es en dicho ámbito donde la comunicación se vuelve un requisito indispensable y son estos dispositivos los que están abriendo un nuevo marco de aplicaciones dentro de la tecnología médica.

Es esto lo que ha llevado a desarrollar soluciones tecnológicas de sistemas *e-Health* (electronic Health), herramientas basadas en las TICs utilizadas en tareas de prevención, diagnóstico, tratamiento, seguimiento, y gestión de la salud y de la forma de vida, que pretenden mejorar la autogestión, toma de decisiones del paciente y en definitiva establecer una mejora en los cuidados de pacientes crónicos. Dentro de este tipo de tecnologías se ubican las conocidas como *m-Health* (mobile Health), que emplean las tecnologías móviles para la mejora en los servicios de salud. La idea de aplicar este tipo de soluciones radica en la reducción de costes sanitarios en atención primaria y la mejora en el acceso al cuidado y atención sanitaria, así como un alivio en la gestión de tareas rutinarias. Principalmente se convierten en un pilar de la autogestión de enfermedades crónicas, como por ejemplo la psoriasis, donde el paciente puede realizarse a sí mismo las tareas de monitorización con su propio móvil, evitando la necesidad de realizar un seguimiento visual por el especialista en el propio centro sanitario bajo cita previa. Siguiendo este ejemplo, las aplicaciones de este tipo evitarían la necesidad de desplazamientos y el empleo de dispositivos sanitarios para tareas que hoy en día se pueden realizar de forma más sencilla y rápida sin pérdida de calidad. Además estos sistemas permitirían la comunicación directa en cualquier momento entre especialista y paciente, si este necesitase una atención crítica.

Uno de los principales problemas que plantea el empleo de estas tecnologías es la interoperabilidad o habilidad de que dos o más sistemas y componentes intercambien información y la utilicen. Para ello se siguen estándares que permiten establecer la comunicación entre los componentes del sistema y tomar un formato de datos reconocible entre los componentes y los usuarios que lo empleen. Los principales estándares que afectarán a este trabajo se pueden agrupar en:

- Interoperabilidad: En imagen se emplea principalmente el estándar DICOM [1], unifica el almacenamiento y comunicación de imágenes médicas en un protocolo único. Este indica que la imagen se almacena junto a una etiqueta que contiene sus datos de tamaño y codificación. Para la comunicación entre los dispositivos electrónicos se tendrá el estándar ISO/IEEE 11073 [2], que proporciona información sobre los datos capturados, información de los signos vitales y datos operacionales del dispositivo. El trabajo emplea una base de datos médica desarrollada en una solución FHIR [3], estándar desarrollado por HL7, cuya flexibilidad permite emplear otros estándares de variados ámbitos y su empleo en diferentes contextos como las aplicaciones móviles o servidores de comunicación. Los datos almacenados en la base de datos tienen que emplear un

sistema de expresiones comprensible por los especialistas médicos, el principal estándar de terminología clínica empleado es el SNOMED [4]. Este recoge la terminología que permite codificar, recuperar, comunicar y analizar los datos clínicos

- Seguridad: La ley orgánica de protección oficial de datos de carácter personal [5] rige las principales características de manejo de datos de pacientes, teniéndose que asegurar el acceso restringido a esta información en la plataforma. Con esto se aplican protocolos de seguridad para la comunicación, como es HTTPS, y estándares de codificación, como DES3 [6], para asegurar la seguridad de la información manejada. Otro estándar empleado en la protección en el manejo de datos es el OAuth[7], que proporciona flujos de autorización para permitir el acceso a los datos al tiempo que se mantienen las credenciales del usuario.

Con estas ideas en mente, el trabajo plantea el desarrollo de una aplicación para la monitorización de la psoriasis. La psoriasis es una enfermedad crónica de la piel debido a una repuesta inflamatoria del sistema inmunitario. Actualmente afecta a un 2,3% de la población y afecta principalmente a personas de entre 15 y 35 años [8]. No es una enfermedad hereditaria pero tiene una cierta predisposición genética y puede ser desencadenada por factores varios. Esta enfermedad se caracteriza por el mal funcionamiento de los linfocitos T, células encargadas de la defensa de nuestros cuerpos, que desencadenan respuestas en la piel a nivel celular. Se producen respuestas de proliferación y dilatación de vasos en la piel y la proliferación celular en la epidermis, lo que provoca el enrojecimiento de la piel y las descamaciones exageradas respectivamente.

Teniendo en cuenta las características de esta enfermedad, su diagnóstico se puede establecer según el tipo de lesión que tenga el paciente [9]:

- Psoriasis en forma de placa: Representa la mayoría de casos de psoriasis diagnosticada. Se caracteriza por la aparición de placas de piel seca con descamaciones blancas sobre la zona que sufre el eritema. Éstas pueden aparecer sobre cualquier parte del cuerpo y de forma simétrica sobre brazos y rodillas.

- Psoriasis en forma de gota: Se observa la aparición de eritemas sobre el tronco y las extremidades, caracterizada por la aparición de enrojecimientos con forma de gota de alrededor de 1cm. Puede acarrear la aparición de infecciones y en un pequeño porcentaje de paciente el empeoramiento hacia psoriasis tipo placa.

- Psoriasis pustulosa: Caracterizada por la aparición de pústulas sobre los eritemas enrojecidos. Las pústulas suelen confluir y formar grandes sábanas de pus. Los pacientes más graves con esta condición suelen estar muy enfermos y requieren de hospitalización con monitorización constante de su estado.

- Psoriasis eritematosa: Se caracteriza por la aparición de las lesiones, anteriormente nombradas, en la mayor parte del cuerpo del paciente. Se da en casos que no se ha vigilado el desarrollo de la psoriasis llegando a un estado en que la vida del paciente puede correr riesgo.

- Psoriasis centrada en determinadas áreas del cuerpo: Como se indica aparecen en áreas concretas como son codos, rodillas, palmas, plantas, genitales, uñas y cuero cabelludo.

En cada persona la psoriasis es diferente y le afecta de forma diferente, por lo que el tratamiento se ajusta a cada paciente. Debido a esto se requiere una monitorización constante de la enfermedad para evaluar su progreso y plan de acción contra ella, siendo una opción disponible la tele-monitorización vía móvil. Aquí es donde el interés de esta plataforma está más marcado, ya que permite el empleo de diferentes microservicios para esta tarea, desde una funcionalidad para recordatorios de revisión y tratamientos hasta la propia funcionalidad desarrollada en este trabajo para monitorizar lesiones.

El trabajo parte de la plataforma de mensajería Signal adoptada por Surya Roca en [10]. Por tanto se tiene una base de aplicación *m-Health* que permite comunicarse a los pacientes con los especialistas y ofrecer a los usuarios una serie de funcionalidades básicas implementadas en estructura de microservicios. La arquitectura de microservicios o MSA establece la construcción de una aplicación como un conjunto de pequeños servicios. Estos servicios son independientes entre sí y pueden comunicarse entre ellos empleando protocolos simples HTTP con REST. Cada microservicio ofrece una funcionalidad diferente y la estructura permite que el lenguaje de programación sea diferente entre ellos. Es debido a la escalabilidad que presenta esta estructura que cada servicio puede ser reemplazado o actualizado de forma independiente, incluso permite la evolución de la misma mediante la integración de nuevos servicios.

Para la interacción autónoma del usuario se ha establecido el empleo de un *bot* o programa autónomo capaz de realizar tareas, que permita al usuario realizar todas las operaciones necesarias desde su dispositivo móvil. Al no disponer Signal de una API para desarrollar un *bot* en la plataforma, éste ha tenido que ser implementado en la plataforma y los microservicios. Será un *bot* de tipo *chatbot* [11], intercambiará mensajes con el usuario desplegando las funcionalidades de la plataforma en una estructura de menú. Dependiendo de la selección del usuario, el *bot* actuará en consecuencia ejecutando las acciones del microservicio integrado.

La aplicación desarrollada en este trabajo se ubica como un microservicio dentro de la plataforma previamente desarrollada, ofreciendo una funcionalidad de procesado de imágenes. La aplicación provee de los recursos para que el paciente pueda enviar una imagen a la aplicación para estudiarla y realizar el seguimiento establecido por el especialista. Esto permitirá a los pacientes que empleen la aplicación disponer de un dispositivo de monitorización, dentro de la aplicación, que les evitará realizar desplazamientos innecesarios al centro médico. Por parte del especialista, esta aplicación le permite realizar una observación previa de la evolución del paciente y establecer un criterio de actuación, pudiendo informar inmediatamente al paciente mediante otras funcionalidades de la plataforma. Con este trabajo se establece una base para el procesado de imágenes en la plataforma de mensajería, que permitirá realizar monitorizaciones de la evolución en las lesiones de psoriasis. La figura 1.1 muestra la arquitectura básica de la plataforma donde se integra la solución para dar funcionalidad.

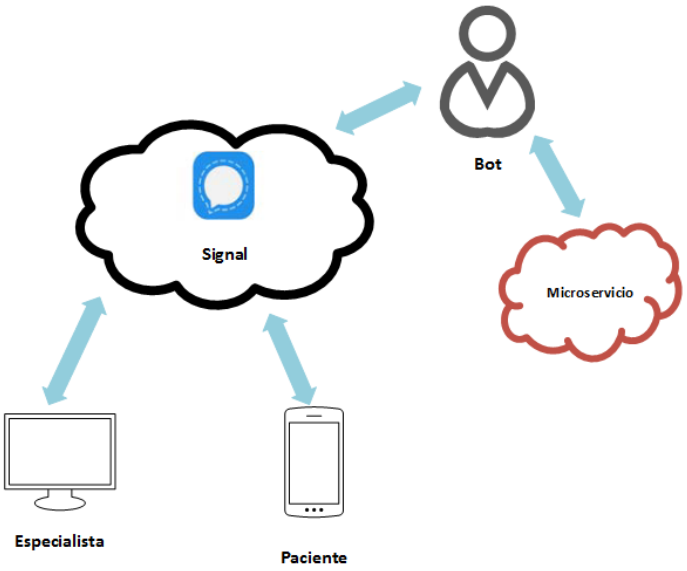


Figura 1.1 Arquitectura básica de la solución integrada en la plataforma

1.1. Objetivos

- Obj1: Desarrollo servidor con *bot* para integrar en la plataforma Signal.
- Obj2: Desarrollo algoritmos de procesado de imagen.

El objetivo principal de este trabajo es desarrollar un microservicio de monitorización de lesiones de psoriasis integrándose en la plataforma de mensajería Signal como una funcionalidad ofrecida a los usuarios por el *bot* de esta plataforma.

Otro objetivo principal del trabajo se centra en el tratamiento y análisis de las imágenes tomadas por los usuarios. Esto permitirá la extracción de información relevante sobre el desarrollo de la psoriasis que pueda ser útil tanto para el paciente como para el especialista.

El *bot* ofrecerá a los usuarios una selección de funcionalidades, entre las que se encontrará la de este microservicio. La propia funcionalidad de la plataforma solicitará una imagen tomada por el usuario para realizar un análisis visual sobre las características de la lesión de psoriasis. Dependiendo de los datos facilitados por el usuario, el análisis se ajustará al tipo de lesión del paciente.

De esta forma se implementarán las funciones necesarias para la interacción entre el usuario y el *bot*, de forma que la monitorización y gestión de información en la aplicación sea completamente autónoma. También se implementarán los distintos métodos y algoritmos de análisis ajustados a las necesidades comunicadas por el paciente.

Los objetivos persiguen la ampliación del servicio iniciado por la plataforma de comunicación previamente desarrollada y que compagine nuevas funciones de análisis de imágenes a las funcionalidades existentes y futuras que pueda albergar.

1.2. Materiales

Para el desarrollo de este trabajo ha sido necesario el empleo de los siguientes materiales, programas y tecnologías.

- Aplicación de mensajería Signal [11]: Aplicación de mensajería ofrecida por la Open Whisper Systems para la comunicación privada simple entre personas. Se emplea en su versión móvil.

- Dispositivo móvil Android: Móvil con sistema operativo Android, para comprobar el funcionamiento del microservicio en la aplicación. Se ha empleado el modelo ZTE BLADE A610. Con cámara trasera de 13MP de resolución, apertura focal 2.2, zoom de 4x aumentos y 30 FPS de máximo de grabación, y con cámara frontal de 5 MP.

- Folio con marcadores: Un folio con cuatro círculos de 4 cm de diámetro, empleado para recortar un área específica de la imagen donde se quiere concentrar el procesado de imagen. Los marcadores se sitúan próximos a las esquinas del folio, situándose a 4 cm de los bordes del folio tanto a lo largo como del ancho del mismo.

- Ordenador: Se ha empleado como medio de programación del microservicio y como servidor provisional para las pruebas de todos los scripts durante su desarrollo. Se ha empleado un portátil HP pavilion g6, con sistema operativo Windows 8.1.

- Ordenador (Servidor plataforma) [10]: Este ordenador en continuo funcionamiento realiza la función de servidor de la plataforma.

- Servidor Signal [11]: Servidor ofrecido por Open Whisper Systems para el funcionamiento de la plataforma de mensajería Signal.

- Geany [12]: Programa multi-lenguaje de edición de código empleado para el desarrollo de los scripts en lenguaje Python del microservicio. La versión empleada es Geany 1.32.

- Idle Python [13]: Herramienta de edición de scripts incluida en la base de Python3.

- Repositorio Github [14]: empleado para almacenar los scripts desarrollados de este proyecto. **Los archivos** corresponden al conjunto que compone al microservicio que está integrado en la plataforma de mensajería y que **se pueden consultar en** [14].

- Virtual Box [15]: Programa de código abierto para la virtualización de sistemas operativos. Se ha empleado la versión Virtual Box 5.2.4 r119785.

- Ubuntu [16]: Sistema operativo libre basado en Linux, empleado con la máquina virtual para elaborar los certificados y llaves de autorización para el microservicio. Se emplea la versión Xubuntu 14.10.

- Debian [17]: Sistema operativo libre basado en Linux, empleado para ubicar el servidor de la plataforma sobre la que se integrará el microservicio. Se emplea la versión 8.8, con arquitectura de 32bits.

- Python [18]: Lenguaje de programación interpretado y multiparadigma de código abierto utilizado para la programación del microservicio.

- Aiml [19,20]: Lenguaje de programación basado en XML, se ha empleado en el desarrollo de la base de conocimientos del *bot* del microservicio.

- Servidor fhir [3]: Base de datos para el almacenado de los datos de los usuarios, sus imágenes, resultados, recodatorios, etc.

- FHIR [3]: Estándar, con bases de estándares HL7, para el intercambio de información médica de forma electrónica, permite intercambiar recursos con estructuras XML o JSON con elementos *wearables* y móviles.

- Opencv [21]: Librería de recursos para la visión por computador en lenguaje Python y Java.

1.3. Organización de la memoria

La memoria se estructura en los siguientes capítulos y anexos:

- **1 Introducción**: Este capítulo recoge, como su nombre indica, la introducción del trabajo desarrollado con las principales bases del mismo junto con los objetivos, materiales y herramientas utilizadas.

- **2 Estructura de la solución**: Este capítulo recoge los principales aspectos del microservicio desarrollado como son su estructura, componentes y funcionamiento.

- **3 Procesado de imagen**: Este capítulo recoge los procedimientos empelados para realizar los tratamientos de imagen necesarios para el análisis de la psoriasis.

- **4 Pruebas de concepto**: Este capítulo recoge las pruebas realizadas para comprobar el funcionamiento de la aplicación desarrollada en este trabajo.

- **5 Conclusiones y líneas futuras**: Este capítulo recoge las conclusiones alcanzadas a lo largo del desarrollo del TFM y las futuras líneas de trabajo que se pueden seguir de él.

- **Anexo A**: Recoge los acrónimos que aparecen a lo largo de esta memoria.

- **Anexo B**: Recoge el proceso de generación de certificados para la comunicación HTTPS.

- **Anexo C**: Recoge el diagrama de intercambio de mensajes entre el usuario de la plataforma y el *bot*.

- **Anexo D**: Recoge la estructura del archivo AIML y la disposición de las variables de operación empleadas para especificar las tareas a realizar.

- **Anexo E**: Recoge la estructura de datos empleada en la base de datos FHIR para su envío y almacenado en esta.

- **Anexo F**: Recoge el estudio realizado sobre diferentes procesos, algoritmos, escalas y métodos que se han ido realizando durante el desarrollo del trabajo y que no han sido o en parte han sido utilizados para establecer el algoritmo final de procesado implantado.

- **Anexo G**: Recoge la definición de operaciones morfológicas y las principales operaciones que se emplean en el trabajo.

1.4. Distribución del tiempo

La distribución del tiempo invertido se puede observar en la tabla 1.1, la cual recoge el diagrama de Gantt del trabajo desarrollado.

La tarea de aprendizaje ha consumido una parte importante del tiempo, debido a la necesidad de asentar las bases de programación de las diversas librerías de Python y el estudio de métodos de procesado de imagen que se ajustasen a las necesidades del proyecto. Esto permitió realizar las funciones de los algoritmos de procesado de imagen y el programa de comunicación del microservico en las tareas de desarrollo de algoritmos e implementación del microservico, respectivamente.

La tarea de configuración se ha solapado entre el desarrollo de algoritmos y la implementación para ajustar las funciones del algoritmo al funcionamiento del *bot* implementado en el microservico. Con esta tarea también se ajusta el programa a los requisitos de la plataforma para la integración.

Se ha realizado una tarea de pruebas para comprobar el correcto funcionamiento de todos los componentes que conforman la aplicación del microservico, así como el funcionamiento una vez está integrado. Con estas pruebas se realiza a la par el rastreo de fallos y aplicación de soluciones. Finalmente, se realiza la tarea de redacción.

Id.	Tarea	Comienzo	Fin	Duración	T4 17			T1 18			T2 18		
					oct.	nov.	dic.	ene.	feb.	mar.	abr.	may.	jun.
1	Aprendizaje	04/10/2017	12/02/2018	94d									
2	Desarrollo algoritmos de tratamiento de imagen	23/10/2017	21/12/2017	44d									
3	Configuración microservicio	04/12/2017	19/01/2018	35d									
4	Implementación microservicio	08/01/2018	09/04/2018	66d									
5	Pruebas de funcionamiento	15/02/2018	15/05/2018	64d									
6	Mejoras y arreglos	09/04/2018	15/05/2018	27d									
7	Redacción memoria	15/05/2018	22/06/2018	29d									

Tabla 1.1 Diagrama de Gantt de la distribución del tiempo

2. Estructura de la solución

En este capítulo se expone la configuración y el desarrollo del microservicio creado. Este microservicio seguirá las especificaciones de desarrollo establecidas en [10], donde se implementa la plataforma de mensajería Signal según la estructura de la figura 2.1. El microservicio irá integrado en las funcionalidades que provee el *bot* de Signal, el cuál recibe peticiones de los usuarios a través de la plataforma de mensajería y los redirige al microservicio solicitado por el usuario.

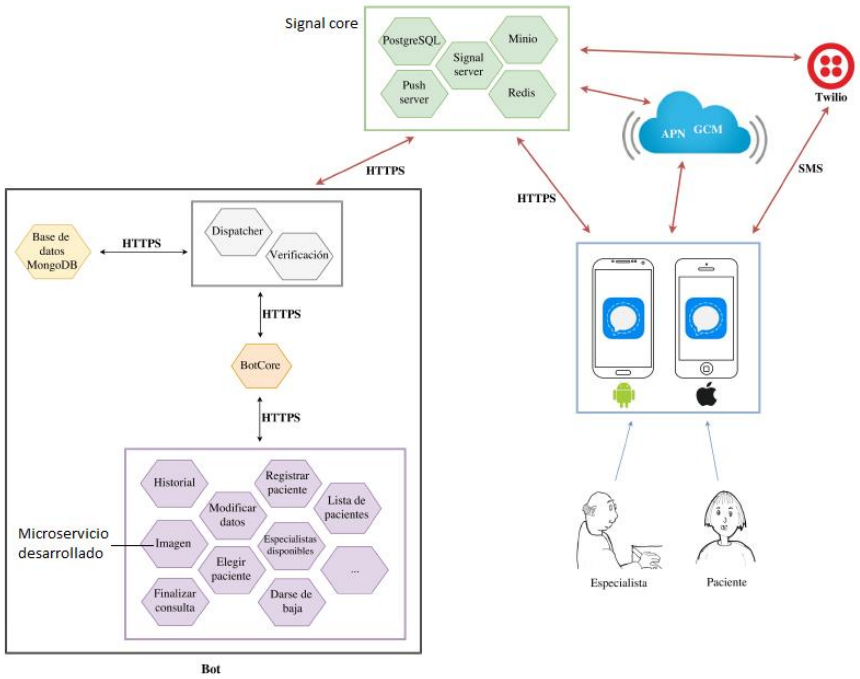


Figura 2.1 Escenario Global [10]

Para que se pueda integrar en la plataforma el microservicio tendrá una estructura de cliente/servidor. Esta estructura le permitirá realizar las tareas de comunicación con los diferentes servicios externos de la plataforma necesarios para que pueda realizar la tarea de monitorización. El servidor implementado para el microservico será el que se encargue de dar funcionalidad mediante la comunicación con la plataforma Signal, mientras que otros recursos de cliente implementados le permitirán gestionar las peticiones a la base de datos médica FHIR, al soporte de autentificación OAuth y al envío de mensajes a la plataforma Signal. De esta forma el microservicio se encarga de todo lo necesario para realizar el procesado desde que recibe la solicitud por mensaje, hasta que almacena y envía los resultados para su visualización.

2.1. Servidor del microservicio

El servidor del microservicio ha sido creado en base a las especificaciones establecidas por la plataforma previamente desarrollada. Se establece de esta forma una estructura en la que el microservicio desarrollado se integra en la plataforma para encargarse de proporcionar una funcionalidad de monitorización de psoriasis a los

usuarios que realicen la petición. De esta forma dispondrán de la opción para enviar una imagen para analizar y registrar la extensión de la lesión.

El servidor Signal está alojado en un servidor físico de la Universidad de Zaragoza, bajo el dominio de "signal.ehealthz.es" y corriendo en una máquina virtual con sistema operativo Debian/Linux. El microservico de procesado de imagen por otro lado estará en la misma red de trabajo que el servidor Signal, ya sea sobre el mismo servidor físico o en otro conectado a la misma red de trabajo, con una IP fija vinculada al dominio "procesado.ehealthz.es".

El lenguaje de programación empleado para el desarrollo del microservico es el lenguaje Python. Dado su carácter de código libre permite emplear gran número de librerías, tanto estándar de Python como desarrolladas por la comunidad, para el desarrollo desde aplicaciones de procesado de imagen como la implementación de servidores y clientes. Una de las características de la plataforma que permite emplear este lenguaje es la estructura API REST, que a través de peticiones HTTP, establece la comunicación de los distintos componentes de la plataforma. La ventaja de esta estructura se hace evidente al poder emplearse cualquier lenguaje de programación pero manteniendo la estructura REST de comunicación, en la que los intermediarios intercambian series de peticiones y respuestas, ya sean de recursos o datos.

2.1.1. Instalación y configuración

Para el funcionamiento del microservicio ha sido necesario instalar y configurar los distintos recursos para el funcionamiento en la plataforma, teniendo en cuenta como están instalados y el proceso que se sigue en [10].

Primero, se requirió la instalación del núcleo de Python, descargado de [18]. Cuya herramienta permitió programar y compilar los distintos scripts que conforman el servidor del microservicio en la aplicación. Seguidamente se continuó con la configuración de los distintos recursos integrados en la plataforma para el reconocimiento del dominio empleado por el microservicio en la comunicación con los mismos. Como unidad cliente, se instaló la aplicación Signal en el móvil para poder realizar los ensayos y pruebas del microservico. Para poder usar la palicación se realizó el registo del usuario móvil en la plataforma rellenando el formulario de solicitud con los datos personales, que se guardaran en el registro de pacientes de la plataforma.

Además ha sido necesario configurar las características de seguridad de la plataforma para el reconocimiento, del servidor aquí desarrollado, como un elemento más del sistema de la plataforma. Para ello fue necesario crear certificados auto-firmados entre el servidor del microservicio y los distintos recursos para establecer una comunicación segura a través del protocolo HTTPS (la creación de los certificados se desarrolla en el anexo B). Por otro lado, el registro en el recurso de expedición de *tokens* permite al servidor solicitar el recurso para obtener *tokens* de autentificación. Con este *token*, el servidor puede autenticarse frente a otros recursos del sistema como perteneciente al mismo, lo que indica que no es un programa extraño o infiltrado y la conexión es sólo entre miembros del sistema dentro de la plataforma.

2.1.2. Estructura

Una vez que el servidor está instalado y configurado sobre el servidor físico de la universidad, dará servicio a los usuarios de la plataforma de mensajería ofreciendo la funcionalidad desarrollada para el microservicio. La plataforma está formada por diferentes elementos y recursos que se comunican con el servidor desarrollado empleando el protocolo de comunicación HTTP, figura 2.2. El servidor del microservicio se comunicará directamente con los elementos sin interactuar con el servidor principal de Signal, por otro lado la comunicación con el usuario móvil se tiene que realizar directamente sobre el servidor Signal ya que es el que gestiona la distribución de la mensajería.

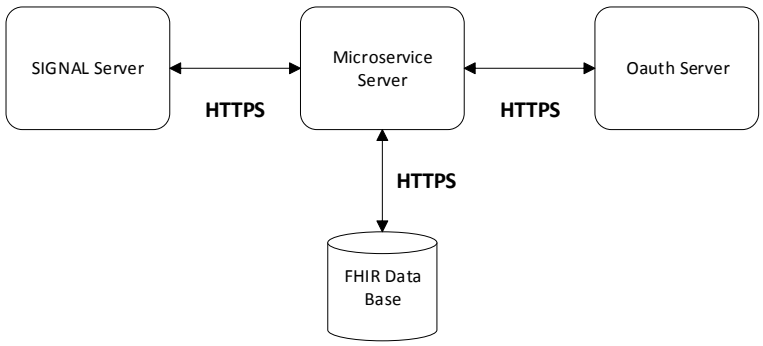


Figura 2.2 Estructura comunicación del servidor

El programa principal del servidor se encarga de la gestión de las peticiones que llegan de los usuarios a través del servidor Signal como mensajes para realizar las operaciones pertinentes. El servidor se comunica a su vez con la base de datos médica FHIR para obtener las imágenes a analizar y de la misma forma almacenar los datos obtenidos del procesado de imagen.

2.1.3. Funcionamiento

Inicialmente, el servidor se arranca y tiene que registrar la funcionalidad en el servidor Signal. Para ello se envía una petición a "signal.ehealthz.es" por el puerto 7001 a la dirección "/new/functionality" con el mensaje 'Procesado_imagen' que indica el nombre de la funcionalidad que se va a registrar (petición POST). De forma similar al apagar el servidor del microservicio, se procede a realizar la eliminación (petición DELETE) del mismo en el registro, se envía el mensaje de la misma forma al servidor Signal pero a la dirección "/delete/functionality" y el mensaje conteniendo el id del microservicio; el mismo introducido anteriormente.

Seguidamente al registro de la funcionalidad, se establece la comunicación con la base de datos FHIR para actualizar el estado del Device a activo. Para ello se realiza una petición PUT a "fhir.ehealthz.es" en el puerto 8080 con un *body* que contenga los datos del estado del Device para actualizar. Con el mismo proceso se actualiza el estado a inactivo cuando se procede a la desconexión del servidor antes de eliminarse la funcionalidad del registro.

Una vez que el servidor está en marcha, éste espera las peticiones de los usuarios enviadas a través de los mensajes que se reciben en el servidor de Signal. El recorrido de los mensajes seguirá la estructura de la figura 2.1, saliendo del móvil al *core* de la plataforma para que el *bot* lo gestione y lo redirija al microservicio que preste la funcionalidad.

Los mensajes enviados por los usuarios pasan a través del servidor de Signal hasta el *bot*, que es el que se encarga de redirigirlos a los diferentes microservicios que ofrecen funcionalidades a la plataforma. Una vez que se solicita la funcionalidad el usuario pasa de estar en comunicación con el *bot* de la plataforma a estar en comunicación con el bot del microservicio. Este *bot* funciona de forma similar al de la plataforma ofreciendo las distintas opciones al usuario para ejecutar las diferentes funciones de análisis que ofrece. Una vez que finaliza las operaciones el usuario, se procede a salir de la funcionalidad volviendo a redirigirse el tráfico de mensajes al *Botcore* de la plataforma. Esto se realiza mediante peticiones a "signal.ehealthz.es" por el puerto 7001, con protocolo HTTPS, para solicitar los datos del usuario a la base de datos. El servidor de la plataforma se encarga de realizar la petición de los datos del usuario solicitado y redirigirlos al microservicio. Con los datos del usuario se establece la funcionalidad 'Procesado_imagen', y se reenvía a Signal con una solicitud de *update* para la base de datos, redirigiéndose así el tráfico de mensajes.

El servidor estará continuamente escuchando los mensajes de las peticiones POST que le llegan a la dirección "/message" y se tomará el *body* del mensaje para realizar la interacción con el *bot*. La interacción con éste seguirá un flujo de trabajo, que se detalla en el apartado 2.2, que establecerá las distintas operaciones a realizar. Cuando se requiera de enviar un mensaje al usuario, se realizará mediante peticiones POST a "signal.ehealthz.es" por el puerto 7001 en las direcciones "/send/message", si se envía solo mensaje de texto, y "/send/attachment" si se envía un imagen con o sin texto. En ambos casos la estructura del mensaje a enviar al servidor Signal para que la reciba el usuario de la plataforma sigue la estructura del código 2.1, donde en los espacios de *attachments* y *body* se introduciran los datos de la imagen y el mensaje, respectivamente, o se establecerán a *null* si no se envían datos de ese tipo.

```
'{"user":"+3469…","platform":"","message":{"timestamp":0,"attachmen
ts":"imagen","body":"texto mensaje"}}'
```

Código 2.1 Ejemplo estructura mensaje json para la comunicación con el usuario [14]

Con los datos del usuario solicitante se prosigue a descargar la imagen enviada por el usuario. Esta se solicita mediante una petición a la base de datos FHIR, donde es alojada por el servidor Signal al enviarse la imagen, mediante la dirección especificada en el *body* del mensaje de solicitud de funcionalidad que llega al microservicio. De esta manera se realiza una petición GET a "fhir.ehealthz.es" en el puerto 8080 en la dirección especificada en el mensaje, "/Media/765" como ejemplo.

Cuando la imagen es descargada, esta es almacenada en la dirección "/imágenes" del servidor dentro de una carpeta con el identificador del usuario en Signal (en este caso el número de teléfono del usuario). En esta carpeta no solo se almacenarán las distintas imágenes generadas durante el análisis, sino también los datos relevantes del análisis y otras variables referentes al procesado, guardándose en el archivo datos.dat (ejemplo de

lista de datos almacenados en tabla 2.1). Este archivo contendrá los datos de las áreas, tanto de los marcadores como de la psoriasis detectada (valores en píxeles), y de la opción escogida para realizar la función de eliminar vello de la imagen.

area_t	tarea_pelo	area_marcador
90234	'No_Pelo'	17028

Tabla 2.1 Ejemplo de datos contenidos en el archivo datos.dat

Con estos datos se realizan las operaciones descritas más adelante en el apartado de procesado de imagen y se almacenan los datos resultantes de la misma manera en un archivo datosDB.dat para ser posteriormente almacenado en la base de datos si lo desea el usuario (ejemplo de lista de datos almacenados en tabla 2.2). Éste contendrá varios campos referentes a las distintas áreas detectadas (en píxeles), los puntos centrales de los contornos detectados de psoriasis y los valores de los pixeles (en escala HSV), para realizar estudios posteriores.

area	puntos	píxeles
[4483, 2009, 389, …, …]	[[456,44], [444,888], [112,990], […], […]]	[[7, 189, 220], [160, 88, 0], [3, 77, 255], […], […]]

Tabla 2.2 Ejemplo de datos contenidos en el archivo datosDB.dat

Una vez que el usuario realiza las operaciones los datos son almacenados mediante una petición POST en la base de datos FHIR, almacenándose como un recurso *Observation*, con la dirección "/Observation" dentro de la base de datos. En este recurso se almacenarán los distintos datos obtenidos y guardados en "datosDB.dat", así como la imagen con los contornos marcados y el área total detectada de psoriasis.

Al terminarse los procesos o cuando el usuario desee salir se procede a realizar un proceso de eliminación de la carpeta del usuario creada en el servidor del microservicio. Este proceso elimina la carpeta, previamente creada al solicitar la funcionalidad, con todo su contenido, incluyendo las imágenes obtenidas del análisis, los datos y la imagen bajada de la base de datos médica FHIR.

Todas estas interacciones del microservicio con los componentes de la plataforma se pueden observar en la figura 2.3 como el desarrollo del intercambio de mensajes que realizan durante el funcionamiento normal del microservicio.

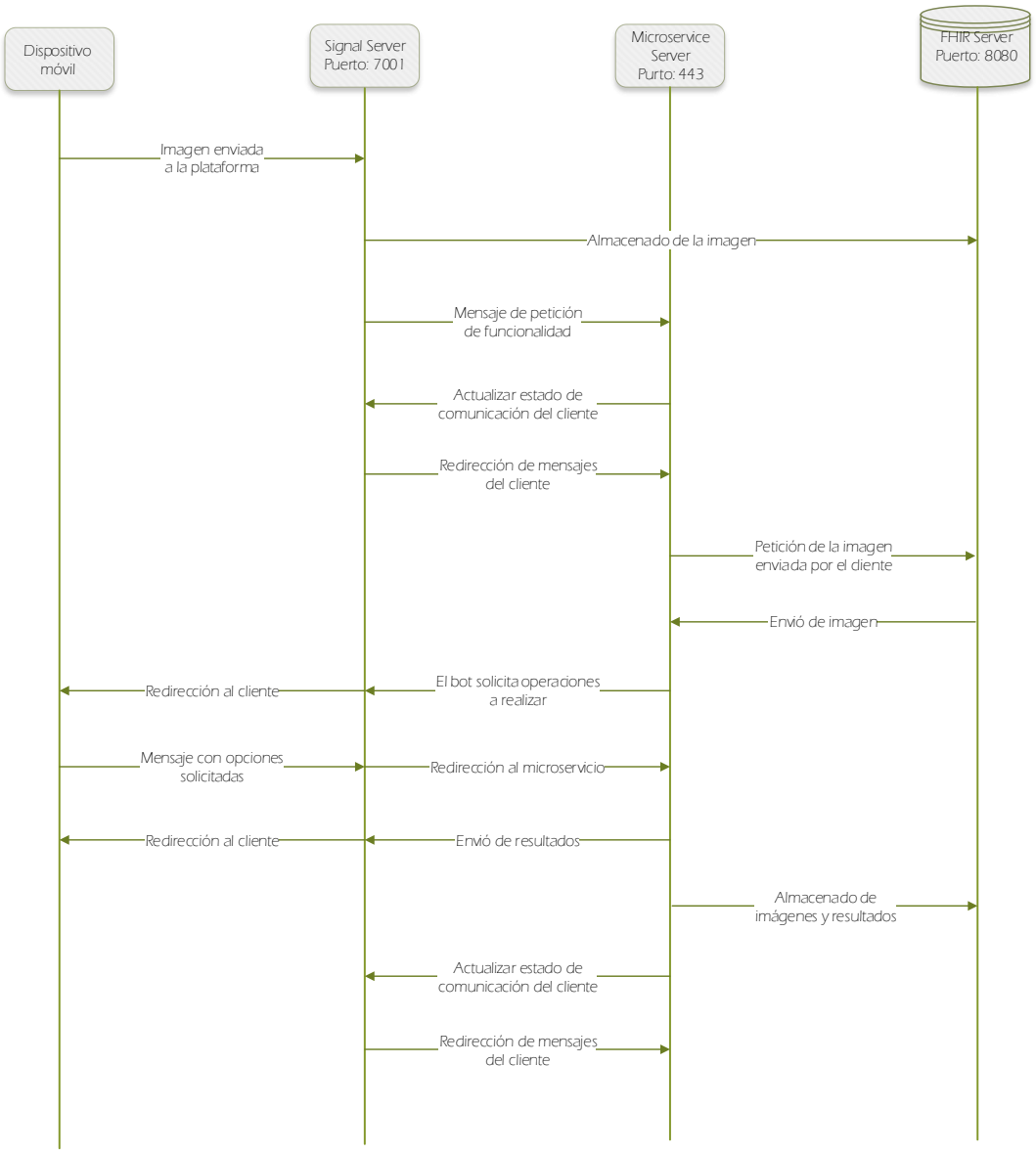


Figura 2.3 Intercambio de mensajes del microservicio

2.1.4. Seguridad

Debido a la naturaleza delicada de los datos que se van a manejar en el servidor se requiere la implantación de ciertas medidas que eviten la interacción indebida y el uso inapropiado de los datos de los usuarios, siguiendo lo establecido en la ley de protección de datos [5] y asegurando que se incluyen los cambios que trae la nueva ley europea de protección de datos [22]. Al tratarse de datos médicos, éstos son cedidos dentro del consentimiento de uso para fines meramente de salud, no se podrán difundir y tendrán que ser almacenados en la base de datos médica con las medidas de seguridad específicas de la base de datos [3].

Puesto que las imágenes tienen que ser tomadas de la base de datos para su tratamiento, éstas están codificadas en base64 y son decodificadas una vez llegan al servidor del microservicio. Tras realizar los tratamientos y análisis, tanto las imágenes como los datos son codificados y almacenados nuevamente en la base de datos médica y

posteriormente son eliminados en el servidor del microservicio. De esta forma, junto al sistema de autentificación mediante *tokens*, nos aseguramos que los datos no puedan ser leídos durante la transferencia de mensajes por agentes externos al sistema. Además la eliminación de los datos ya almacenados y guardados evita la posible distribución indebida de los datos personales, en este caso imágenes.

Como se explica en el apartado anterior, las imágenes para tratar son almacenadas en una carpeta del servidor. Aunque para la comunicación del servidor ha sido necesario programar las acciones a realizar cuando recibe las peticiones POST únicamente, se ha programado la petición GET, para fines de desarrollo y futuras aplicaciones, que podría permitir el acceso a las imágenes por parte de cualquier solicitante. Esto rompe la norma y establece la necesidad de implantar una limitación en las peticiones. Para evitar el acceso a las direcciones de los datos e imágenes se ha introducido en la carpeta "imagenes" un archivo tipo ".htaccess" que restringe el acceso de cualquier IP que no sea la del propio servidor, código 2.2.

```
<Limit GET>
 Order Deny,Allow
 Deny from all
 Allow from 192.168.100.10
</Limit>
```

Código 2.2 Código del archivo image_secure.htacces [14]

La seguridad en la comunicación entre elementos del sistema se ha desarrollado de varias formas. Como se ha nombrado varias veces previamente, todos los microservicios, recursos y otros elementos integrantes del sistema de la plataforma se reconocen entre ellos mediante el uso de *tokens* de verificación. Siempre que se realice alguna comunicación se estable un campo "Authorization" en la cabecera para el *token*, código 2.3. Una vez que el mensaje llega se toma el *token* y se verifica su procedencia en el recurso de verificación de la plataforma (servidor Oath). En caso de que no sea válido, se interrumpe la comunicación con un mensaje de error de autentificación al solicitante.

```
token = tokens.server_token().get_tokenDB()
headers = {'Content-Type': 'application/json',
           'Authorization': 'Bearer ' + token['Data']['id_token'],
           'Content-length': str(len(payload))}
```

Código 2.3 Fragmento de código de cabecera con token [14]

La comunicación HTTP empleada por el microservicio entabla el riesgo de que alguien pueda interceptar los mensajes de la comunicación establecida. Con la implementación de una comunicación HTTPS empleando el protocolo TLS1 se resuelve este problema. Para implementar esta comunicación se tuvieron que desarrollar claves y certificados que permitieran a los integrantes de la plataforma establecer una comunicación cifrada con el servidor del microservicio. Esto restringe la comunicación a elementos que dispongan del certificado que asegura la comunicación segura a través

del canal, con la clave de codificación que dispone el servidor del microservicio. La creación de la clave y los certificados se desarrollan en el anexo B.

2.2. Bot

Como medio para la interacción del usuario con la aplicación se ha establecido el uso de un *bot* que permita dar autonomía a los usuarios de la aplicación a la hora de realizar tareas médicas rutinarias, como monitorizaciones y gestiones de recetas o formularios médicos. Éste despliega ante el usuario las distintas opciones disponibles en forma de menús con las opciones que permitan realizar las funciones desarrolladas en los microservicios e interactúa con los usuarios mediante intercambios de mensajes [22].

El *bot* de la plataforma Signal se encarga de la distribución de los mensajes y de la llamada a las funcionalidades solicitadas por los usuarios. Como se comenta anteriormente, el tráfico de mensajes es redirigido al microservicio de la funcionalidad una vez esta es solicitada y el *bot* de la plataforma deja de contestar a los mensajes que llegan. El manejo de estos mensajes por parte del microservicio requiere el desarrollo de un *bot* tipo *chatbot* que los gestione. De esta forma al recibir un mensaje, éste se pasa al *bot* que emite una respuesta en consecuencia que es enviada al usuario, siguiendo un esquema de interacción como el de la figura 2.4.

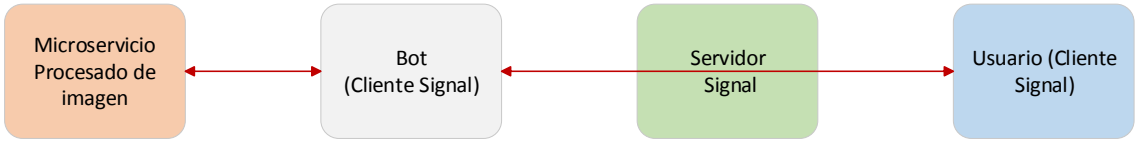


Figura 2.4 Escenario de interacción con bot

2.2.1. Estructura y funcionamiento

El *bot* desarrollado está integrado en el servidor del microservicio recibiendo los mensajes de los usuarios que son redirigidos por la plataforma. Una vez que el mensaje es enviado al *bot*, el usuario contestará en consecuencia y el servidor realizará tanto las operaciones específicas como el envió del mensaje al usuario. Por su parte el usuario contestará a las peticiones del *bot* que permitan realizar las operaciones según los requisitos especificados. En el diagrama de uso de la figura 2.5 se puede apreciar el escenario antes nombrado con los dos actores, el *bot* y el usuario. En éste se puede apreciar la solicitud de funcionalidad por parte del usuario a la plataforma para que el *bot* le provea del servicio, junto con los resultados de operación, y los requisitos de operación que deben ser facilitados por el usuario.

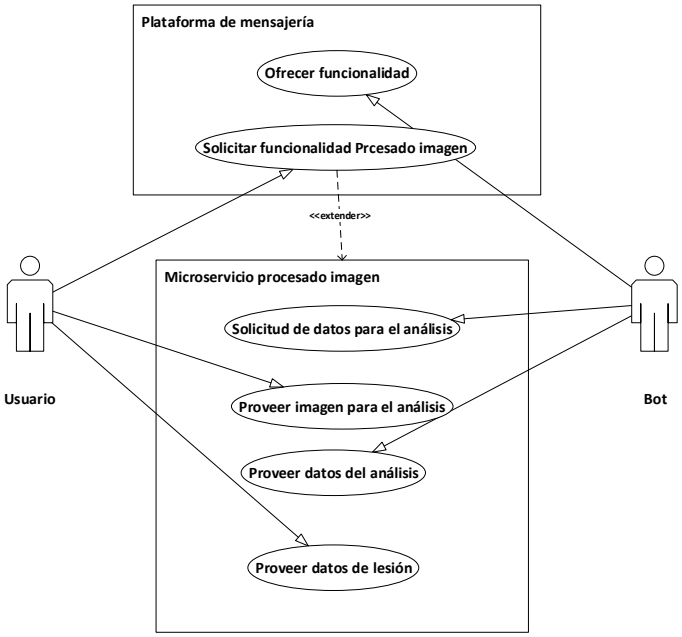


Figura 2.5 Diagrama de uso

Para la interacción se ha desarrollado un *script* en lenguaje AIML ("interaccionbot.aiml") [19], este lenguaje se basa en la estructura XML y recoge los conocimientos que permiten al *chatbot* responder a las contestaciones del usuario. El *script* desarrollado sigue el esquema de conversación de intercambio de mensajes del anexo C. En éste se establecen las respuestas a enviar al usuario y las señales de realización de los procesos, tales como el guardado de datos en la base de datos y los procesos de procesado de imagen. Por lo tanto este esquema de interacción está fuertemente relacionado con el flujo de trabajo del procesado de la imagen, figura 3.1.

El *bot* ha sido programado en lenguaje Python empleando la librería estándar "Python-aiml" [20]. Ésta permite de forma rápida y simple la implementación de un *bot* mediante comandos sencillos que engloban desde la recpción de frases al bot, hasta las funciones que permiten al bot aprender las respuestas programadas en los archivos AIML. Como ejemplo el código 2.4 representa la programación requerida para implementar cualquier *bot* con ésta librería.

```
import aiml
kernel = aiml.Kernel()      #Kernel bot
kernel.learn('archivo.aiml')
while True:
        resp = kernel.respond(input("Enter your message >> "))
```

Código 2.4 Ejemplo de comandos para implementación de bot [14]

El proceso que se sigue para la implementación se inicia con la creación del *kernel*, o nucleo de procesado que se encargará de reconocer las respuestas a las preguntas recibidas, como ya se ha dicho el archivo AIML contiene las respuestas programas que serán las que emplee el *kernel*. El comando "respond" es el que permite introducir las frases al *bot* y recibir la consiguiente respuesta. En el servidor las selecciones del usuario introducidas al *bot* no solo devuelven la respuesta, sino que también devuelven las señales de ejecución de tareas que permiten al servidor realizar los procesos cuando es solicitado por el usuario. Como se observa en el código 2.5, el *script* "bot.py"

desarrollado introduce los mensajes del usuario al *bot* y devuelve la respuesta junto con las variables de operación. La variable 'microservice' indica la operación para la redirección de los mensajes al microservcio y la variable 'tarea' indica las operaciones de procesado a realizar para el análisis. El programa desarrollado y la disposición del archivo AIML para obtener variables de operación, se desarrolla más en detalle en el anexo D.

```
[responsebot, microservice, tarea] = bot.response(aux[0])
```

Código 2.5 Fragmento del código de "servidor.py" referente al bot [14]

2.3. Almacenamiento de datos

Para el almacenamiento de la información se emplean distintas bases de datos. Por un lado están las bases de datos que maneja la plataforma para la gestión de usuarios y la mensajería [10] y por otro lado la base de datos médicos FHIR, que almacenará toda la información médica que emplee la plataforma. El servidor de este trabajo no manejará directamente las bases de datos que gestionan los usuarios y la mensajería, como se ha explicado anteriormente, para solicitar alguna información del usuario o cambiar el estado de comunicación con los microservicios se realizan peticiones al servidor Signal y este gestionará las operaciones. En cambio como el servidor desarrollado manejará información sobre la monitorización y el análisis visual, éste se encargará del guardado de estos datos y la breve gestión que estos conllevan.

Dada la organización de la base de datos FHIR [3], ésta puede contener una gran cantidad de información relevante. Tanto la información de los pacientes como de los especialistas es generada previamente una vez que se ha producido el registro en la plataforma. De forma similar algunos campos han tenido que ser creados previamente para su empleo en operaciones posteriores. En el servidor desarrollado se ha tenido que crear y gestionar la información del campo "Device" que recoge la información relevante que indica el instrumento o aparato que ha sido empleado en alguna operación o tarea. Así mismo, en el campo de "DeviceMetric" se establece la información referente a las mediciones realizadas por el "Device" y se conecta a éste para referenciar que son las medidas que emplea. Más detalle de los datos almacenados del "Device" y del "DeviceMetric" en los códigos E.1 y E.2 del anexo E.

Al realizar las operaciones solicitadas de procesado de imagen, se requiere que al almacenar los datos se indique la petición realizada para este proceso. Por ello, se ha tenido que establecer la información de los campos de "DeviceRequest", para indicar el empleo del dispositivo por parte del paciente, y el "ProcedureRequest", para indicar el proceso que se ha llevado a cabo (monitorización como procedimiento para el caso de este trabajo). Más detalle de los datos almacenados del "DeviceRequest" y del "ProcedureRequest" en los códigos E.3 y E.4 del anexo E.

Ya con esta información disponible para realizar las operaciones, lo primero que se realiza es la actualización de los datos de la imagen enviada por el usuario a la plataforma, para indicar el "Device" empleado. Seguidamente la imagen es descargada del servidor y se opera con ella. Los datos e imagen obtenidas son almacenadas en el

campo de “Observation”, donde se referencian con los campos previamente creados y el paciente para tener toda la información enlazada referente al proceso. Más detalles de los datos almacenados en “Observation en el código E.5 del anexo E.

Con los datos almacenados de esta manera, otros microservicios pueden extraer en conjunto los datos y realizar estudios sobre la evolución de la psoriasis que pueda ayudar a los especialistas en la toma de decisiones. Además, como se comentaba anteriormente, se almacenan los valores de los pixeles de las áreas detectadas, lo que permite hacer un seguimiento de los falsos positivos para detectar y realizar estudios de ajuste y mejora de los algoritmos de detección y análisis.

3. Procesado de imágenes

En este capítulo se van a exponer las distintas técnicas empleadas en el procesado de las imágenes enviadas por los usuarios a la plataforma. La aplicación está orientada para la monitorización de las lesiones de psoriasis, ésta recogerá la información visual del área de lesión. Los datos obtenidos permitirán al especialista realizar un seguimiento de la evolución de la lesión. Esto entablará el primer acercamiento de la plataforma en ofrecer recursos más complejos que ayuden en la observación y diagnóstico del especialista.

Como herramienta de desarrollo de los algoritmos de procesado se han utilizado los recursos disponibles en la librería OpenCV [24,25]. Esta librería de código abierto optimizada para el procesado de imagen se establece como una de las principales opciones en Python y otros lenguajes como estándar en procesado, tanto en el ámbito comercial como el académico.

El estudio previo de las características de la psoriasis ha llevado a establecer un proceso de detección en el que se comprueba la diferencia de tonalidades de la piel para detectar las áreas de psoriasis. Como se expresa en [9,26], una forma de abordar la detección de psoriasis es mediante la extracción de características de las tonalidades rojas de la imagen, siendo las todos rojos de la lesión los que más destaquen de esta forma. De forma similar se puede hacer con los tonos blancos de la placa que forma la psoriasis, aunque según las investigaciones no suele ser tan efectivo como el método de estudio del enrojecimiento es lo suficientemente efectivo como para establecer una detección de áreas blancas en la imagen. Otra característica que se puede emplear en la detección de la psoriasis es el aumento en el grosor de la piel. Esta característica, aunque difícil de manejar en análisis de imágenes, nos permite realizar un acercamiento mediante la detección de bordes que resalten en la imagen. Con esto en mente se establece un proceso de análisis basado en la detección de áreas rojas, blancas también en caso de que haya placa de psoriasis, y como apoyo la detección de bordes para marcar los cambios sobre la piel que produce la psoriasis.

A continuación se expondrán los distintos métodos implementados que sigue el procesado. Estos seguirán el flujo de trabajo establecido por el *bot* de forma que siga el esquema de la figura 3.1. A parte durante el desarrollo del trabajo se han implementado y estudiado diferentes algoritmos y métodos de detección antes de que se estableciera el desarrollo final. Estos estudios se desarrollan en el anexo F.

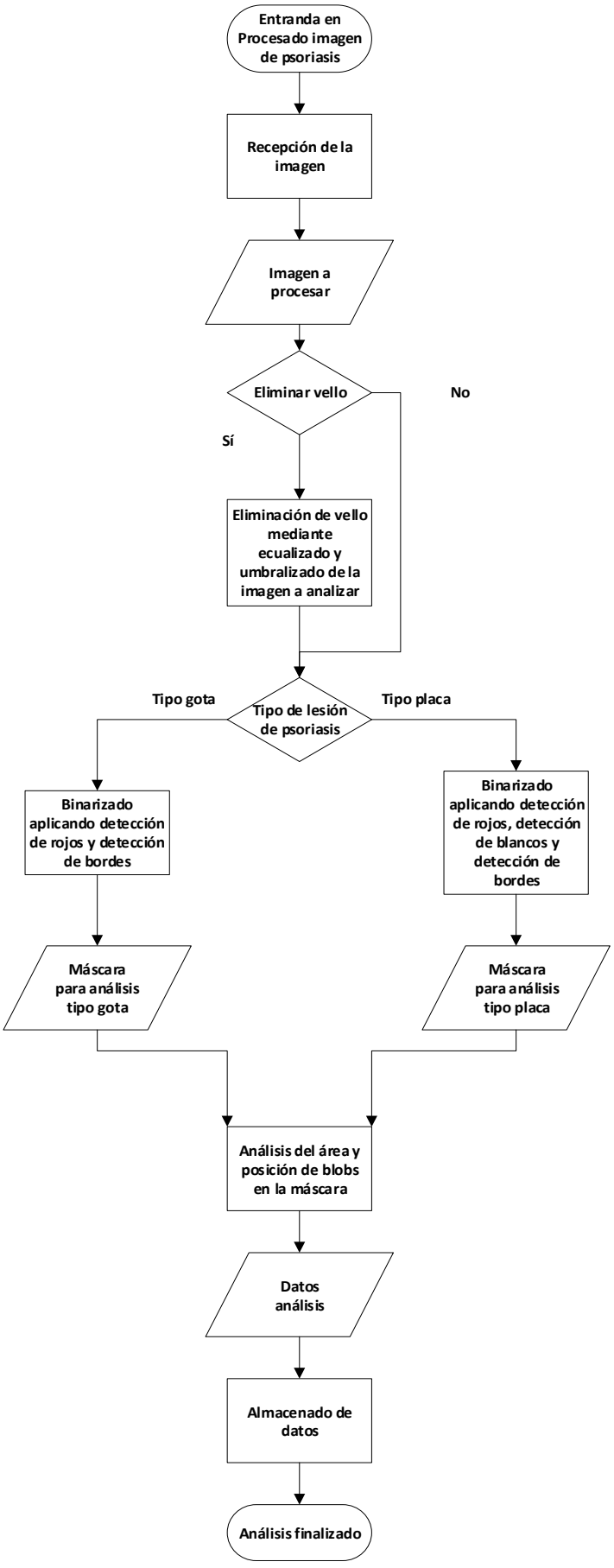


Figura 3.1 Diagrama de flujo de trabajo

3.1. Limpieza y preparación de la imagen

Se describen a continuación los procesos que permiten obtener una mejora en la calidad de la imagen para, por tanto, mejorar la detección de la psoriasis. Como paso inicial en cualquier proceso de tratamiento de imagen, se han realizado operaciones de filtrado previas a la extracción de características. Para este trabajo se ha empleado un filtrado Gaussiano que elimine el ruido que se pueda introducir en la imagen. Teniendo en cuenta la buena calidad de las imágenes realizadas con el móvil y el que la aplicación sea un prototipo han llevado a implementar el filtro pero sin que se aplique en el programa ahorrando en procesado.

3.1.1. Estudio filtrado

Según el procedimiento habitual de procesado de una imagen, uno de los primeros pasos es realizar un filtrado que elimine el ruido que contenga la imagen. Para este trabajo se ha implementado un filtrado Gaussiano que permita eliminar el posible ruido que contenga la imagen, figura 3.2.

Figura 3.2 Filtrado gaussiano, imagen original (izquierda) imagen filtrada (derecha)

Las imágenes obtenidas por el móvil ofrecen una alta calidad en la imagen y la posibilidad de repetir el proceso evitando tener que trabajar obligatoriamente con una imagen que contenga ruido, borrones o en general de baja calidad. Esto junto con la poca diferencia obtenida en los resultados de detección, figura 3.2, llevaron a estimar el ahorro en procesado del proceso de filtrado, pero manteniéndolo en el programa si es necesario para futuros usos.

Como se observa en las imágenes de la figura 3.3, la detección tiende a variar poco con o sin filtro, pero varía de imagen a imagen. En el primer caso se observa que empeora la detección perdiéndose gran cantidad de información, en el segundo caso de prueba la detección es similar pero se pierden algunos positivos de la detección, el último caso muestra que la detección mejora ligeramente y se reduce la detección de pelo. Este patrón se repite en sucesivas pruebas donde apenas se obtiene mejora y lleva a la decisión de no aplicar el filtrado implementado.

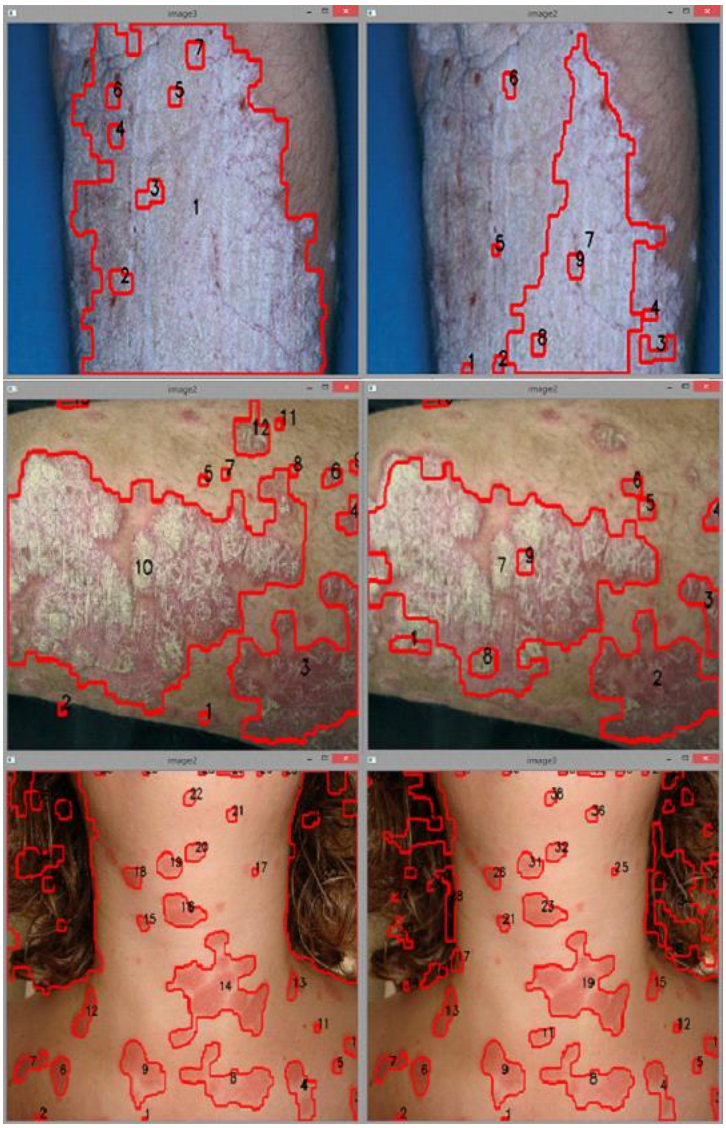


Figura 3.3 Pruebas con filtrado, sin filtrado (imágenes izquierda) con filtrado (imágenes derecha)

3.1.2. Recorte por marcadores

Para estudiar un área concreta de la imagen, se ha implementado un *script* que permite recortar una zona delimitada por cuatro marcadores. La imagen es transformada a escala de grises y se umbralizan los valores altos de contrate para obtener una máscara en la que resalten los marcadores. Los marcadores podrán ser de color negro sobre fondo blanco o de color blanco sobre fondo negro, siempre guardando el alto contraste para facilitar la detección de los mismos. Seguidamente, la detección de los marcadores en la máscara se realiza mediante la transformada de Hough para la detección de círculos [27]. Este método emplea la fórmula de la circunferencia:

$$(x - a)^2 + (y - b)^2 = r^2$$

donde se supone que (x, y) están fijos en la imagen y se tiene que buscar el centro (a, b) y el radio r. Se buscan todas las posibles figuras y se guardan sus valores en un acumulador, donde se comprueba los valores máximos que marcan la figura buscada. Conociendo los círculos y sus centros, se recorta la imagen siguiendo los puntos. La

imagen resultante sería similar a la obtenida en la figura 3.4, a la que también se le eliminan los marcadores pintando círculos negros sobre los detectados. De esta forma se reduce la interferencia que pudieran causar en los subsiguientes procesos de detección.

Figura 3.4 Imagen inicial (izquierda) imagen recortada (derecha)

3.1.3. Eliminación de vello

Uno de los problemas más destacados que se observan en la detección es el elevado número de falsos positivos que aparecen en la imagen a consecuencia del vello. Esto se debe a que cuanto más oscuro es el vello más influye en las tonalidades rojas que se estudian para detectar la psoriasis y, por otro lado, se tiene que de forma similar afecta en la detección de bordes debido a que resaltan en la imagen. Esto llevó a tomar la decisión de desarrollar un procesado de imagen que elimine el máximo vello posible para permitir mejorar la detección.

Se ha implementado un script que permite realizar la detección del vello empleando un método simplificado que se aproxima a los empleados en [28] y [29]. El primer paso se centraría en la detección del vello, que se caracteriza por estar fuertemente resaltado en las tonalidades rojas de la imagen. Por lo tanto se extrae el canal rojo para aplicar una ecualización CLAHE que permita resaltar el vello con respecto al resto de la piel. Este método de ecualizado se caracteriza por ser adaptativo, limitando los límites de los valores de la ecualización para que no resalte el ruido en la imagen de salida. Se puede observar este proceso en los histogramas de las imágenes en la figura 3.5. Sobre la imagen ecualizada se aplica un umbralizado adaptativo que detecta el vello de la imagen.

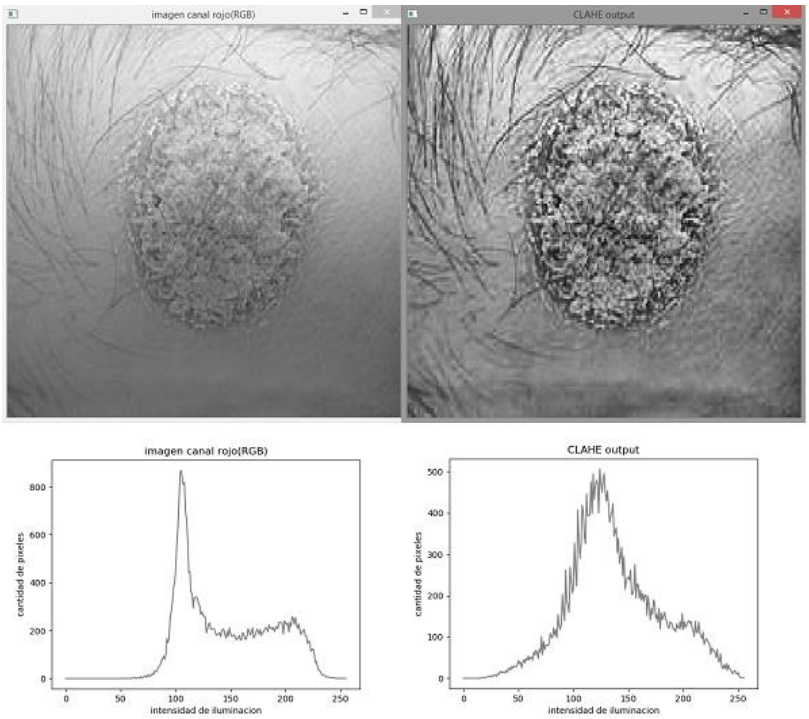


Figura 3.5 Imágenes e histogramas CLAHE

Con la máscara obtenida el segundo paso es eliminar el vello detectado de la imagen original. Con los elementos eliminados sólo queda rellenar esos espacios con color mediante la ponderación con los valores de los pixeles vecinos. El resultado final se puede observar en la figura 3.6 donde el vello ha desaparecido o se ve más atenuado tras el procesado.

Figura 3.6 Resultado al eliminar vello, imagen original (izquierda) imagen de salida (derecha)

3.2. Extracción de características de la imagen

Como previamente se ha ido comentando en el trabajo, el programa desarrollado debe detectar la psoriasis y extraer el área que esta abarca en la superficie de la piel. La detección se centrará en las características principales de las lesiones de psoriasis más comunes, psoriasis tipo gota y psoriasis tipo placa. Según las características de estos tipos de psoriasis se plantea la detección centrándose en las tonalidades rojas de la lesión y en las tonalidades blancas que destaca en la formación de placa. Como apoyo a

estos métodos de detección se ha implementado otro sistema para detectar la psoriasis mediante umbralizado adaptativo, que se basa en los cambios bruscos de color entre la piel y la lesión en la escala de grises y también en la influencia de la rugosidad que produce la placa sobre la superficie de la piel.

Con el estudio realizado sobre los distintos métodos de detección de piel y el empleo de diferentes espacios de color [30,31,32,33,34,35], se tomó la decisión de emplear el espacio de color HSV para la detección de las tonalidades del eritema. Este espacio de color permite manipular: H (Hue) que maneja el matiz de color, S (Saturation) maneja el brillo que va del blanco al negro, V (Value) el valor del color entre blanco y negro. Según el estudio se tendrá en cuenta que las tonalidades de la piel (para Caucasicos y Asiaticos) van entre H 0 a 50 y S entre 0.2 a 0.7, por lo que se evitará que influyan estos rangos en la detección de color.

3.2.1. Detección de rojos

Para la detección de los tonos rojos se sigue un proceso como el del esquema de la figura 3.7. Para realizar la detección se necesita introducir al programa una imagen en escala RGB para su conversión a HSV y sobre esta escala realizar el proceso de detección en base a las características de la psoriasis.

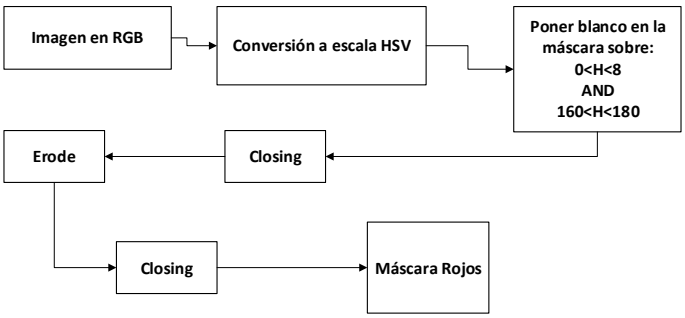


Figura 3.7 Esquema detección tonos rojos

Debido a la lesión que produce el eritema, la diferencia de tonos de piel permite establecer una fuerte diferenciación entre lo que es piel sana y piel de lesión. Se toman los colores rojos en la franja $0<H<8$ y $160<H<180$ y se ajusta $65<S<255$ para evitar detectar los brillos que se pueden dar sobre las tonalidades de la piel sana. Con esto se obtiene una máscara que indica en blanco las zonas rojas detectadas. Como paso final, se hace pasar una serie de operaciones morfológicas (explicadas más en detalle en el anexo G) de *closing* y erosión para obtener la máscara limpia [36].

3.2.2. Detección de blancos

Para detectar los tonos blancos se sigue un proceso similar al de los tonos rojos como se puede ver en el esquema de la figura 3.8. Se inicia con una imagen en escala RGB y de la misma forma se transforma a escala HSV. La detección también es similar estableciéndose los umbrales de detección y realizando los procesos morfológicos en la máscara para su limpieza. El problema que se plantea en la detección de los colores blancos radica en que al aplicar los límites sobre $180<V<255$ se toman todos los tonos blancos de todos los matices y brillos. Por ello se tienen que realizar ajustes sobre el

brillo y el matiz para evitar tomar los colores puros. Se ajusta 0<S<60 para tomar las regiones menor pureza de color. Por otro lado, el matiz se debería observar al completo, pero tras varias pruebas se ha tenido que ajustar sobre 15<H<175 (eliminando los tonos rojos de la piel) al obtenerse una gran detección sobre piel sana de tono muy claro.

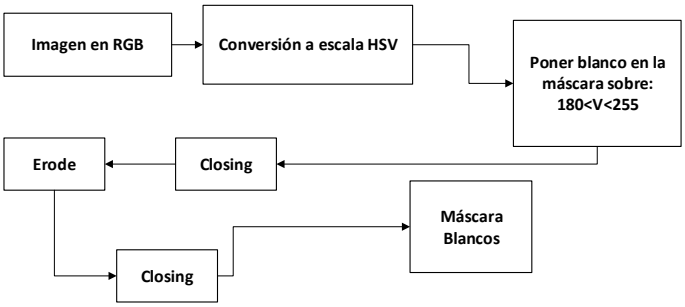


Figura 3.8 Esquema detección tonos blancos

3.2.3. Detección por umbralización

Este método de detección implementado sigue el esquema de operación de la figura 3.9. A diferencia de los algoritmos anteriores requiere convertir a la escala de grises la imagen para poder operar sobre ella y realizar la detección mediante umbralizado. La detección mediante umbralización fue el primer método implementado y probado para la detección de la psoriasis durante el desarrollo de este trabajo, con la idea de detectar los cambios de color en el contraste de una imagen en escala de grises. El problema de emplear esta detección se centra en que no solo reconoce las diferencias de tono de la psoriasis, sino que también reconoce otras imperfecciones de la piel que tengan un alto contraste, como puede ser pelo, pecas, granos, etc. Por otro lado la detección de bordes mediante umbralizado resulta muy útil para detectar cambios que destaquen en la superficie de la piel, como los que produce la placa psoriática por rugosidad o incremento del grosos de la piel.

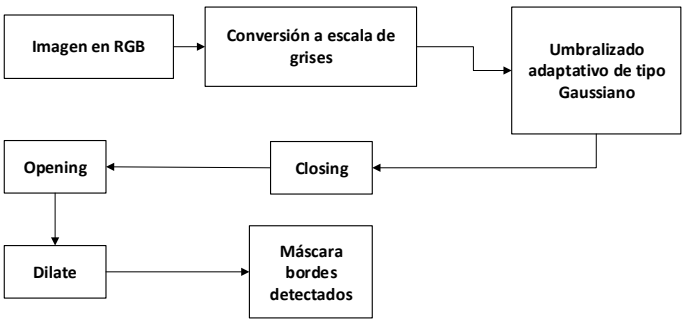


Figura 3.9 Esquema detección mediante umbralizado adaptativo

3.2.4. Detección de contornos

Este es el paso final realizado en la detección de la psoriasis que proporciona la información para ser almacenada en la base de datos y la imagen con los contornos remarcados de las lesiones. Este proceso sigue los pasos del esquema de operación

figura 3.10. El primer paso a realizar, previo a la localización de los contornos, es la unión de las máscaras en una sola que permita realizar la operación de detección de contornos. Esta unión de máscaras viene determinada por el tipo de psoriasis que se quiera estudiar, siendo esta información facilitada por el usuario en la interacción con el *bot*. De esta forma se establecen dos cursos de acción:

- Psoriasis tipo gota: Teniendo en cuenta las características de este tipo de psoriasis, la detección de tonos blancos no resulta efectiva al no haber placa, por lo que no se emplea esta máscara en el procesado. Esto también evita la introducción de ruido en la máscara final que pueda producir falsos positivos. La máscara final se creará a partir de las máscaras de tonos rojos y del umbralizado.

- Psoriasis tipo placa: A diferencia del caso anterior éste emplea las tres máscaras para formar la final. El empleo de la detección de blancos en este caso puede llegar a ser crítico, ya que en los peores casos de psoriasis de este tipo la placa cubre un área mayor que el eritema.

Una vez se tiene la máscara final, se aplica una detección de los contornos blancos, devolviendo los contornos a aplicar sobre la imagen inicial. De esta forma se obtiene la psoriasis de la imagen delimitada por el contorno obtenido.

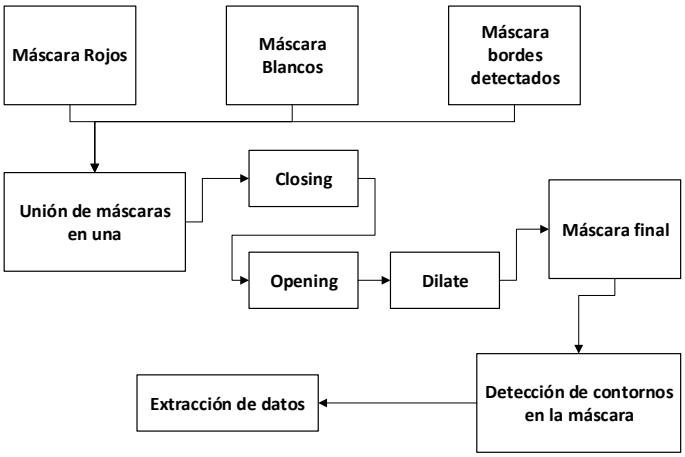


Figura 3.10 Esquema de unión de máscaras y detección de contornos

3.3. Parámetros obtenidos en el procesado

El objetivo de la monitorización es establecer la evolución de la psoriasis en el tiempo como referencia para el especialista. La evolución se medirá mediante el estudio del área detectada en las imágenes facilitadas por el usuario a la plataforma para su análisis mediante procesado de imagen. A parte se tienen que obtener los datos que permitan identificar la posición en la imagen de las lesiones detectadas y extraer otros datos que puedan ser relevantes para futuras mejoras de la aplicación.

Con el objetivo de obtener el área superficial para la monitorización, se realizan los pasos anteriormente descritos para obtener los contornos que engloben la lesión en la imagen. Es con estos contornos que se obtiene el área en pixeles de la lesión que permite obtener el área en cm^2que cubre. Como el área de los marcadores es conocida y

se extrae su área en pixeles, estos se pueden emplear en la estimación de área real de lesión. Para evitar tener que realizar trasformaciones de planos en la imagen y cálculos de geometrías, se puede considerar que tanto la superficie de la piel como el marcador están en el mismo plano de la imagen, con un mínimo margen de error. Esta simplificación implica que con los datos conocidos del área total que cubre la psoriasis en la imagen se puede estimar:

$$Área\ lesión\ cm^2 = \frac{Área\ lesión\ pixels * Área\ marcador\ cm^2}{Área\ marcador\ pixels}$$

Una vez que se extrae el área, con los contornos conocidos se puede extraer más información que caracterice a las lesiones de la imagen. Calculando el punto central de los contornos se puede establecer su posición en la imagen para su identificación. Otro dato que se extrae del contorno son los valores de los pixeles, los cuales permiten estudiar los falsos positivos obtenidos para futuras mejoras del algoritmo de detección.

Al final los datos obtenidos del procesado son: área total de la lesión en cm^2, imagen con contornos detectados, las áreas de los distintos contornos en píxeles, los puntos centrales que identifican a cada contorno y los valores de color en HSV de los pixeles centrales de cada contorno. Finalmente solo queda su almacenado que se ha explicado en el apartado 2.2.

4. Prueba de concepto

En este capítulo se desarrollan las pruebas de concepto para mostrar la viabilidad del microservicio dentro de la plataforma de mensajería Signal, ofreciendo la funcionalidad de monitorización de psoriasis. Estas pruebas permiten ver el funcionamiento de la aplicación dentro de la plataforma de mensajería donde estará implementada, así como el aspecto que tendrá al ser ejecutada por los usuarios en sus móviles, centrándose en la interacción del *bot* con el usuario y como los resultados son mostrados al usuario. De la misma forma se muestra un proceso de ejemplo de los pasos realizados durante el procesado desde que la imagen es recibida y se aplican las distintas operaciones hasta que se marcan los contornos y se obtiene el área total afectada. También se desarrollan una serie de pruebas para el ajuste de la detección de la psoriasis exponiendo las dificultades halladas y los problemas que se plantean para futuras continuaciones.

4.1. Interacción con el usuario

A continuación se desarrolla lo que sería una interacción cualquiera con un usuario de la plataforma. El usuario simplemente tiene que seguir las instrucciones que va desplegando el *bot* en forma de menús para ir completando el proceso de toma de datos de la monitorización. Esta prueba de ejemplo muestra la viabilidad de funcionamiento de la aplicación una vez está implantada sobre la propia plataforma Signal.

Primero, para comprobar si es posible ejecutar la función en plataforma, se comprueba que la conexión del microservio con ésta se realiza correctamente. La propia plataforma nos permite comprobar las funciones disponibles en la pantalla de gestor de acciones a modo de menú de selección, figura 4.1.

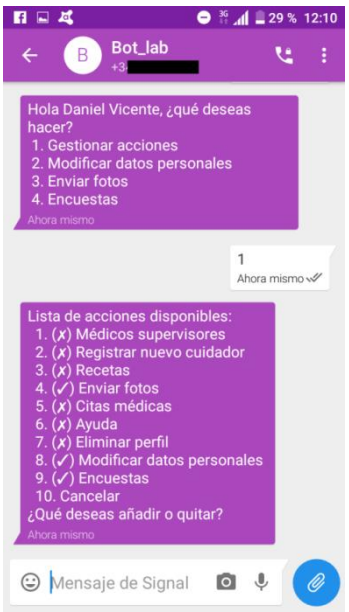


Figura 4.1 Lista de acciones en la plataforma

Se puede observar que la función Enviar fotos, que será la que realizará el microservico desarrollado en este trabajo, está marcada con un *tick* que indica que la

funcionalidad puede seleccionarse y por tanto que el microservicio está conectado. Con la funcionalidad disponible, sólo hay que seleccionarla y realizar la operación que el *bot* nos solicitará realizar, de esta forma se pasa a estar dentro de la funcionalidad del microservicio, figura 4.2.

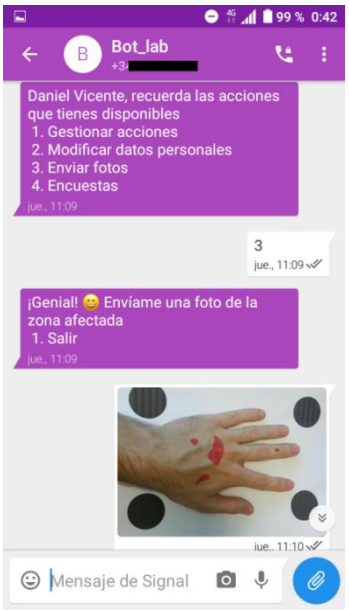


Figura 4.2 Envío de imagen a la plataforma

Una vez la imagen es enviada, ésta es recortada por el rectángulo que forman los marcadores detectados. El microservicio devuelve el recorte para que el usuario pueda verificar que la operación se ha realizado exitosamente. Seguidamente el *bot* enviará las cuestiones que permitirán realizar las operaciones expresadas en el esquema de la figura 3.1. Esta parte de la interacción correspondería a la figura 4.3.

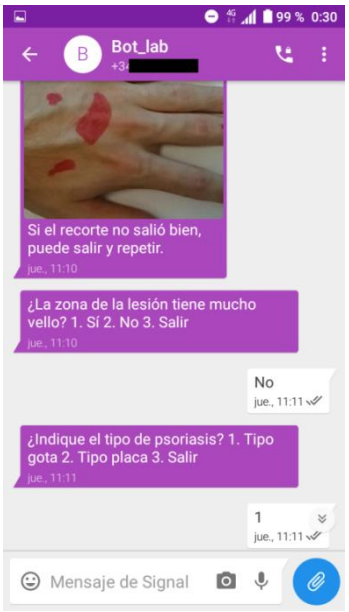


Figura 4.3 Interacción de operaciones de procesado

Como se desarrolla en los apartados anteriores, el microservicio realiza las distintas tareas de procesado ajustándose a las necesidades del usuario, que ha ido contestando a las cuestiones que se le plantean. Primero se le preguntará si la zona tiene mucho vello, el cual puede producir ruido en el procesado, para aplicar o no la operación que lo elimine. Finalmente pregunta el tipo de psoriasis, gota o placa, para realizar el tratamiento y análisis que permita obtener el área de lesión según el tipo de psoriasis que padece el usuario. De esta forma una vez terminadas las operaciones de procesado el microservicio muestra por mensajes en la aplicación los resultados obtenidos con los contornos sobre la imagen y el tamaño estimado de superficie afectada. En la figura 4.4 se puede observar el mensaje que recibe el usuario.

Figura 4.4 Disposición de resultados para el usuario en la plataforma

Una vez los datos son mostrados al usuario, este puede elegir entre repetir el procesado, guardar los datos o salir si lo desea sin que se guarden los resultados. Si se elige repetir, el *bot* repetirá la pregunta del vello y del tipo de psoriasis para hacer un nuevo análisis con las especificaciones nuevas. Si se elige guardar, el microservicio almacena los datos del área obtenida junto con la imagen con los contornos marcados de la psoriasis y sale de la funcionalidad volviendo al menú de selección de la plataforma. Por otro lado la opción de salir, que se puede ejecutar en cualquier momento durante la ejecución de la funcionalidad, saliendo directamente al menú de selección y sin guardar los resultados en la base de datos.

4.2. Procesado de la imagen

En el apartado 4.1 se muestra una interacción patrón entre un usuario cualquiera y el *bot*, donde el usuario responde según sus necesidades de ajuste del procesado. De esta forma, como se comentaba con anterioridad, la aplicación ejecuta las operaciones del diagrama de flujo de la figura 3.1 en relación a las contestaciones del usuario en la interacción. A continuación se expone un ejemplo de los procesos explicados en el capítulo 3 de los procesos de detección que realiza la aplicación sobre una imagen enviada a la plataforma y que no son visibles para el usuario de la aplicación mientras se

están realizando. Este apartado muestra entonces la viabilidad de los distintos procesos aplicados sobre la imagen y sus resultados apreciables en la detección final.

Nada más que la imagen es recibida ésta es recortada siguiendo el proceso del apartado desarrollado en el apartado 3.1.2 y se procede con la primera opción de la interacción. En el caso que el usuario necesite aplicar la eliminación de vello, en la primera opción de la interacción, tendrá que indicar que sí lo desea aplicar. Es entonces cuando se aplicará o no el proceso del apartado 3.1.3 y se pasará a aplicar el procesado de detección de psoriasis con la elección del tipo de psoriasis elegida por el usuario seguidamente.

Con la imagen preparada para realizar las operaciones, se aplica la detección de tonos rojos, explicada en el apartado 3.2.1, para obtener la máscara con las áreas de lesión marcadas en color blanco. Esto permite establecer las zonas de la imagen con los eritemas producidos por la psoriasis que destacan sobre los tonos de piel sana. Como se puede observar en el resultado de la figura 4.5, las zonas marcadas de la máscara indican la lesión de psoriasis. Uno de los problemas que más se observa en estos procesos de detección es la iluminación. Dependiendo de ésta el resultado de la detección se ve fuertemente afectado, ya que, como se puede apreciar en la figura 4.5, las regiones de la pierna mejor iluminadas presentan una detección mejor que las zonas más oscurecidas.

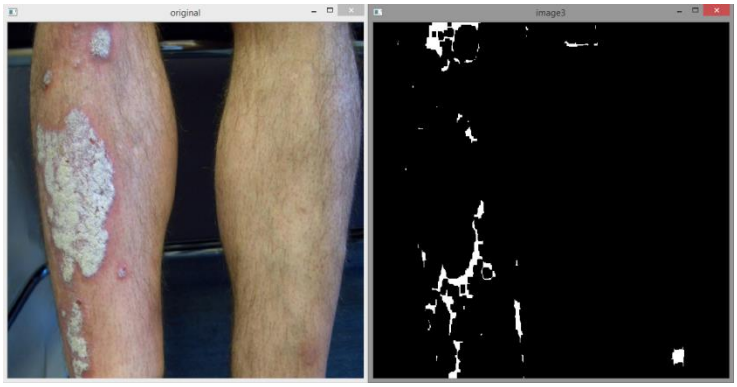


Figura 4.5 Comparación imagen original (izquierda) con máscara de rojos obtenida (derecha)

Para el tipo de lesión de placa se añade una detección más al proceso de operación que permite detectar los tonos blancos de las descamaciones, explicada en el apartado 3.2.2. Debido a que en los estados más avanzados de este tipo de psoriasis se produce éste fenómeno, la detección de los eritemas se vuelve ineficiente para detectar el área de lesión, ya que la mayor parte de esta puede verse cubierta por descamaciones blancas. Uno de los problemas más destacados de este tipo de detección se puede observar en la máscara de la figura 4.6, donde las zonas con un alto brillo se ven marcadas en la máscara. Ciertas regiones de piel siguen siendo detectadas debido al tono claro de la pigmentación y el reflejo que produce sobre esta la luz de la iluminación. Es uno de los métodos más difíciles de ajustar, pues como se expresa en los artículos [9] y [26] no hay un buen método de detección del blanco, debido en gran parte a los problemas nombrados. Una opción es establecer otro método de apoyo que se centre en otra característica observable de la psoriasis como puede ser la rugosidad y grosor en la piel, esto llevó a incluir el método de detección de bordes por umbralizado.

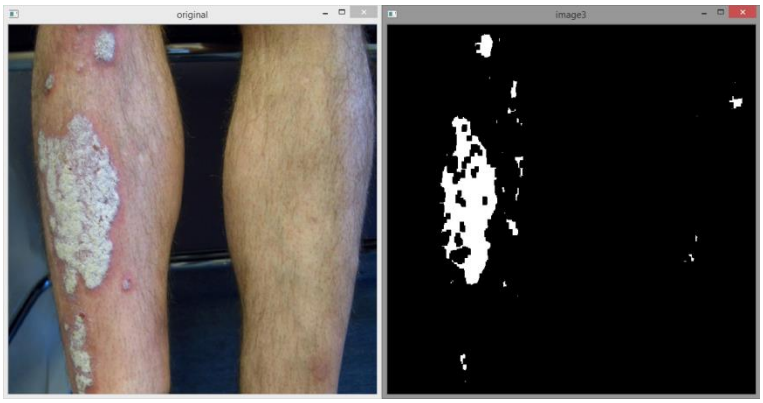


Figura 4.6 Comparación imagen original (izquierda) con máscara de blancos obtenida (derecha)

La detección de bordes, explicada en el apartado 3.2.3, permite detectar las rugosidades, grosores y cambios significativos de color en la imagen. Esto permite obtener una máscara de la lesión pero con una precisión inferior a la que se puede obtener con los métodos anteriores. Debido a que no depende únicamente del color como los métodos anteriores, éste se emplea como apoyo para asegurar una detección más óptima de la lesión. Una de las desventajas que se comentaba de este método y que se ha tenido en consideración, es la alta concentración de ruido que puede generarse en la máscara. En la figura 4.7 se puede observar los resultados de la detección. Como se observa en la detección del umbralizado, la placa es detectada debido al cambio que se aprecia en la superficie de la piel por la rugosidad. Por otro lado, como se comentaba previamente, se observa que las imperfecciones de la piel (en este caso destaca la detección del vello) también son detectadas. Estas detecciones no deseadas se ven fuertemente atenuadas tras el paso de aplicar las operaciones morfológicas sobre la máscara.

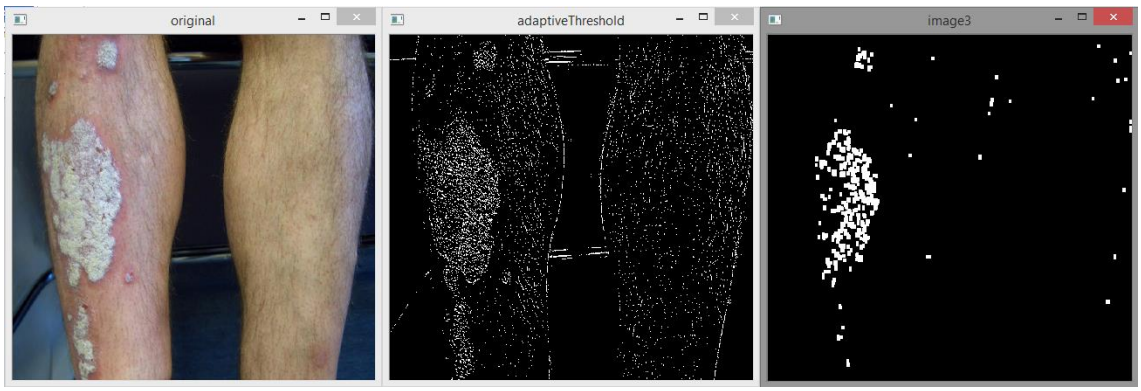


Figura 4.7 Comparación imagen original (izquierda) con detección umbral adaptativo (centro) y mascara obtenida (derecha)

Ya con las máscaras obtenidas sólo queda el proceso final de obtención de los contornos, explicado en el apartado 3.2.4, donde se unen las máscaras empleadas, según el tipo de psoriasis elegido por el paciente en la interacción, para formar la máscara final de la lesión detectada. En la figura 4.8 se puede observar la máscara y el contorno dibujado sobre la imagen original para ser enviada al móvil del usuario. Es con estos contornos con los que posteriormente podemos obtener los valores del área real aproximada de lesión.

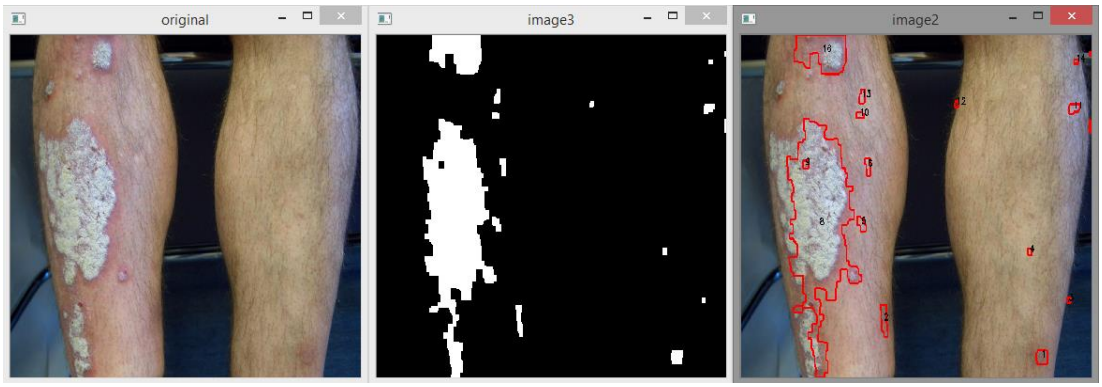


Figura 4.8 Comparación imagen original (izquierda) con máscara detección de psoriasis (centro) y contornos de psoriasis detectados (derecha)

4.3. Problemas y limitaciones

A lo largo del desarrollo de este trabajo se han ido realizando una serie de pruebas que han permitido obtener el ajuste de detección más óptimo posible y la comprobación del correcto funcionamiento de la aplicación. Estas pruebas han permitido establecer las limitaciones y problemas que se han tenido que ir solucionando o atenuando en caso de no poder resolverse. A continuación se enumeran los principales problemas que han tenido que ser enfrentados a lo largo de estas pruebas:

- Detección de vello como psoriasis: El vello se presenta como un problema de ruido en la máscara de la detección que se puede atenuar mediante el algoritmo desarrollado en el apartado 3.1.3. Una limitación que presenta este algoritmo es que en imágenes con demasiada densidad de vello, no alcanza a realizar una eliminación lo suficientemente óptima como para que no sea detectado. En la figura 4.9 se puede observar el resultado de detección en un caso de vello excesivo. Una solución a esta limitación sería desarrollar un algoritmo más sofisticado para la eliminación de vello.

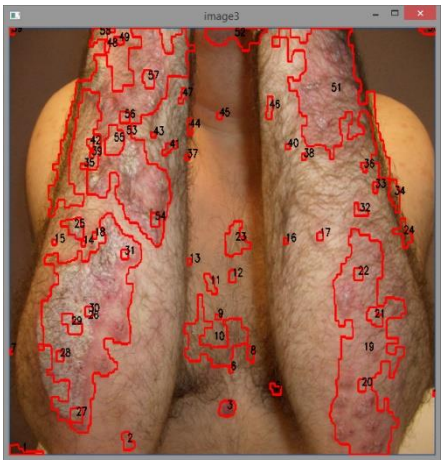


Figura 4.9 Detección con alta concentración de falsas detecciones por el vello

- Problemas de la iluminación en la detección: Uno de los problemas que se plantea en los algoritmos de detección es el cambio en la iluminación y el brillo. Debido a que un cambio significativo en la iluminación afecta a la saturación del

brillo y el valor del color del blanco al negro. De esta forma tras realizar numerosos ajustes en la detección del color, el algoritmo se ve limitado por la calidad de la iluminación, de forma que si la imagen es muy oscura se detectarán más zonas rojas erróneas y si la imagen tiene demasiada iluminación se detectarán más zonas blancas erróneas. Como un ejemplo se tiene la detección realizada en la figura 4.10, donde en la zona superior izquierda de la imagen se observa una pérdida de precisión en la detección por la sombra que oscurece la piel. Debido a esta limitación se decidió extraer el valor de los pixeles detectados para poder realizar ajustes en el algoritmo de la detección. Otra forma de hacer frente a esta limitación sería desarrollar nuevos métodos de detección más complejos.

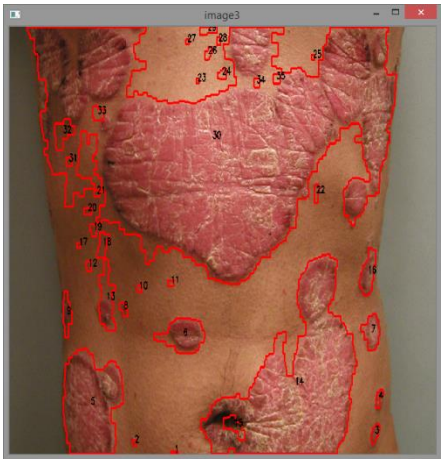


Figura 4.10 Ejemplo de detección afectada por la iluminación

- Área de los marcadores blancos: Una vez que la se obtienen los contornos de psoriasis y se calcula el área afectada, se presenta el problema de que la detección de tonos blancos detecta los tonos blancos del folio. Para evitar este problema se usaron marcadores blancos sobre fondo negro reduciendo así la detección del fondo a solo los marcadores. Los cuatro marcadores se presentan en la imagen recortada como cuatro cuartos de círculo en las esquinas de la imagen, ejemplo en figura 4.11. Como el área real de un marcador es conocida, se puede eliminar esta del área total en cm^2 que se ha detectado en la imagen para obtener una aproximación más precisa.

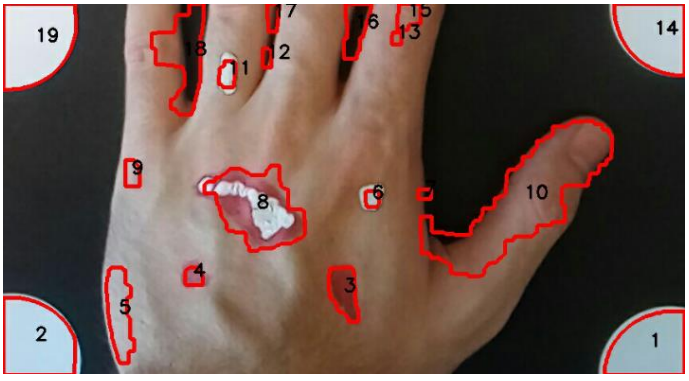


Figura 4.11 Marcadores detectados en recorte.

- Cambio de la resolución de las imágenes al ser almacenadas: Un problema que se plantea en este punto es que si la imagen es de alta resolución (3120x4160, con la cámara del móvil empleado) la base de datos la devuelve con la mitad de resolución (1560x2080). Esto es debido a la normalización que se realiza en la base de datos de las imágenes introducidas a una misma resolución y ahorrar espacio de memoria. Al haber una menor resolución de píxeles, se da que aparecen con más frecuencia agrupaciones mayores de píxeles con valores similares en la imagen. Esto provoca que al realizar las operaciones de detección se detecten zonas que en alta resolución no se detectarían. Un ejemplo es la figura 4.12 en la que se detecta la línea blanca del folio de los marcadores con baja resolución, pero es eliminada como ruido en la detección con alta resolución. Debido a esto cuando se realicen ajustes de detección en el programa en futuros trabajos, se tendrá que tener en cuenta únicamente el efecto que tendrá sobre las imágenes una vez estén almacenadas en la base de datos FHIR.

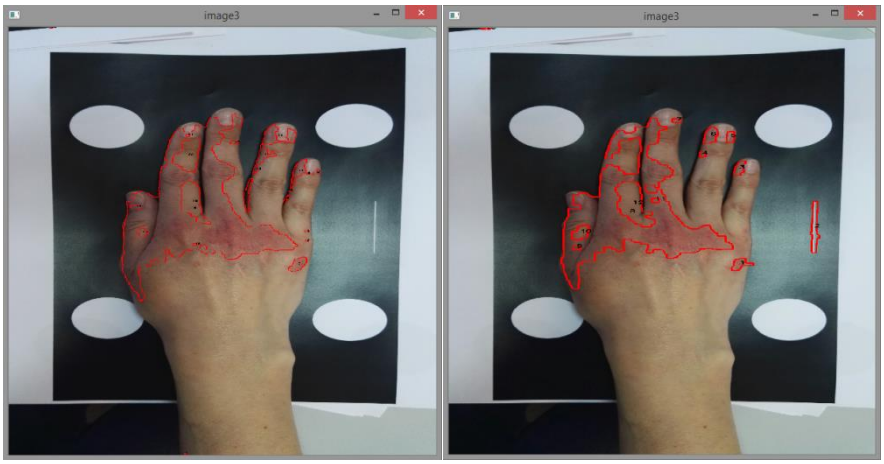


Figura 4.12 Comparación detección con imagen en alta resolución (izquierda) y con resolución reducida (derecha)

5. Conclusiones y líneas futuras

5.1. Conclusiones

Como resultado final, los objetivos del apartado 1.1 han sido completados llegándose a alcanzar una finalización del 95% de los mismos. Los problemas expuestos en el capítulo anterior han provocado que no se pueda llegar a completar al cien por cien los objetivos y han abierto nuevas líneas de trabajo futuras que permitan ampliar lo que se ha desarrollado y completar los pequeños resquicios que pudieran quedar.

Se ha conseguido integrar un microservicio para monitorizar lesiones de psoriasis en la plataforma de comunicación Signal. La estructura en microservicios de la plataforma ha permito que se pueda realizar la implementación de la solución de forma independiente sin que afectase a los microservicios ya implementados en la plataforma. Únicamente se ha tenido en cuenta los requisitos que había que cumplir para poder realizar la integración y ajustar el programa.

Los estudios realizados sobre la detección de la psoriasis han permitido desarrollar las distintas funciones que conforman el algoritmo de detección, siendo uno de los objetivos principales alcanzados por el trabajo. Se han tenido que desarrollar distintos métodos de detección y realizar numerosos estudios para alcanzar este objetivo, llegando a dejar sin implementar funciones que no cumplían con los requisitos deseados o debido a su complejidad resultaban ineficientes.

El trabajo ha requerido realizar el estudio previo de la plataforma Signal ya implementada, para tener en cuenta los requisitos necesarios a cumplir en la configuración del funcionamiento del microservicio. También el estudio de conocimientos relativos a los *bots*, ya que ha sido necesario el desarrollo de un *bot* integrado en el microservicio para interactuar con el usuario, evitando así la intervención de un agente externo y continuando con el *bot* ya implementado en la plataforma.

En la puesta en marcha de la plataforma cabe destacar los numerosos problemas encontrados al realizar la comunicación entre los componentes de la plataforma. Esto sumado a los cambios que ha ido sufriendo durante el desarrollo del trabajo la plataforma, han llevado a numerosos cambios en la configuración para ajustarse a la evolución progresiva que sigue el conjunto del proyecto. Por otro lado las pruebas sucesivas han dado un resultado satisfactorio una vez se alcanzó el resultado final.

A nivel personal, ha sido una experiencia bastante positiva. Han sido muchas horas de trabajo, frustración y resolución de problemas que me han permitido ampliar conocimientos que desconocía, como el diagnóstico de la enfermedad psoriasis, y asentar aquellos que eran solo la base, como los conocimientos de la programación en Python. Este TFM ha servido como reto personal, ya que me ha permitido estudiar más en detalle las plataformas de mensajería y otros elementos de telecomunicaciones con los que no estaba familiarizado en mis estudios de grado.

5.2. Líneas futuras

Se proponen una serie de trabajos futuros que abarcan desde la mejora de la detección de psoriasis al desarrollo de microservicios que puedan utilizar la información obtenida en el análisis de las imágenes para fines que añadan más funcionalidades a la plataforma.

- Añadir nuevos tipos de psoriasis a la detección y el desarrollo de las técnicas que ello implica.

- Aumentar la información relevante obtenida del análisis (actualmente solo área) y almacenarla en la base de datos para uso en estudios de psoriasis.

- Desarrollar un microservicio que emplee la información del área detectada en la base de datos para establecer una gráfica con la evolución del área de la psoriasis a lo largo del seguimiento.

- Crear otros microservicios para monitorizar otro tipo de enfermedades tomando como base el de este trabajo.

- Mejorar las técnicas de tratamiento de imagen aplicadas en el microservicio para conseguir mejorar la detección de la psoriasis.

- Investigación de nuevas técnicas de detección que puedan sustituir alguna de las empleadas en el microservicio, mejorando así su funcionalidad.

- Aumentar la información introducida en la base de datos sobre la información técnica del software desarrollado y de sus características.

- Introducir en la base de datos la información que se pueda obtener de la cámara que toma la imagen para facilitar más información de estudio.

Bibliografía

[1] Estandar DICOM. Accedido en 2018.
https://www.dicomstandard.org/

[2] ISO/IEEE 11073. Accedido en 2018.
http://www.11073.org/

[3] FHIR v3.0.1. Accedido en 2018.
https://www.hl7.org/fhir/

[4] Estándar SNOMED. Accedido en 2018.
https://www.snomed.org/

[5] Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. Accedido en 2018.
http://noticias.juridicas.com/base_datos/Admin/lo15-1999.html

[6] Estándar de encriptación DES3. Accedido en 2018.
https://en.wikipedia.org/wiki/Triple_DES

[7] Estándar OAuth. Accedido en 2018.
https://oauth.net/

[8] Acción Psoriasis. Accedido en 2018.
https://www.accionpsoriasis.org/sobre-la-psoriasis.html

[9] Raychaudhuri, S. K., Maverakis, E., & Raychaudhuri, S. P. (2014). Diagnosis and classification of psoriasis. *Autoimmunity reviews*, *13*(4-5), 490-495.

[10] Surya Roca Mainer. (2017). Diseño de una plataforma de seguimiento y ayuda a pacientes crónicos mediante el uso de bots autónomos basada en la plataforma de mensajería Signal. Universidad de Zaragoza, Campus Río Ebro.

[11] Signal. Accedido en 2018.
https://signal.org/

[12] Geany. Accedido en 2018.
https://www.geany.org/

[13] Guido van Rossum and the Python developmente team. December 15, 2017. The Python Library *Reference Release 3.6.4rc1*. Python Software Foundation.

[14]Repositiorio Github. Accedido en 2018.
https://github.com/DanielVic/micro_psoriasis

[15] Virtual Box. Accedido en 2018.
https://www.virtualbox.org/

[16] Ubuntu. Accedido en 2018.

https://www.ubuntu.com/

[17] Configurar certificados digitales en Debian. Accedido 2018. https://codificate.wordpress.com/2014/11/20/configurar-certificados-digitales-en-debian/

[18] Welcome to Python.org. Accedido en 2018. https://www.python.org/

[19] AIML 2.0 Reference. Accedido 2018. http://callmom.pandorabots.com/static/reference/

[20] Python-aiml. Accedido 2018. https://github.com/paulovn/python-aiml

[21] OpenCV library. Accedido en 2018. https://opencv.org/

[22] GDPR. Accedido en 2018. https://www.eugdpr.org/

[23] Alesanco, Á., Sancho, J., Gilaberte, Y., Abarca, E., & García, J. (2017). Bots in messaging platforms, a new paradigm in healthcare delivery: application to custom prescription in dermatology. In *EMBEC & NBC 2017* (pp. 185-188). Springer, Singapore.

[24] Alexander Mordvintsev and Abid K . September 24, 2017. OpenCV-Python Tutorials Documentation *Release1*.

[25] The OpenCV Reference Manual *Realease 3.0.0-dev*. June 25, 2014.

[26] Shrivastava, V. K., Londhe, N. D., Sonawane, R. S., & Suri, J. S. (2015). First review on psoriasis severity risk stratification: An engineering perspective. *Computers in biology and medicine*, *63*, 52-63.

[27] Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, *15*(1), 11-15.

[28] George, Y., Aldeen, M., & Garnavi, R. (2015, November). Skin hair removal for 2D psoriasis images. In *Digital Image Computing: Techniques and Applications (DICTA), 2015 International Conference on* (pp. 1-8). IEEE.

[29] Çayır, S., & Yetik, I. S. (2017, July). Hair and bare skin discrimination for laser-assisted hair removal systems. In *Engineering in Medicine and Biology Society (EMBC), 2017 39th Annual International Conference of the IEEE* (pp. 604-607). IEEE.

[30] Zortea, M., Flores, E., & Scharcanski, J. (2017). A simple weighted thresholding method for the segmentation of pigmented skin lesions in macroscopic images. *Pattern Recognition*, *64*, 92-104.

[31] Kolkur, S., Kalbande, D., Shimpi, P., Bapat, C., & Jatakia, J. (2017). Human Skin Detection Using RGB, HSV and YCbCr Color Models. *arXiv preprint arXiv:1708.02694*.

[32] Ramello, P. M. (2005). Comparación de métodos de detección de piel en modelos de color YCbCr y HSI para reconocimiento de caras.

[33] Prasetyo, E., Adityo, R. D., Suciati, N., & Fatichah, C. (2017, July). Mango leaf image segmentation on HSV and YCbCr color spaces using Otsu thresholding. In *Science and Technology-Computer (ICST), 2017 3rd International Conference on* (pp. 99-103). IEEE.

[34] Soriano, M., Martinkauppi, B., Huovinen, S., & Laaksonen, M. (2000). Skin detection in video under changing illumination conditions. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on* (Vol. 1, pp. 839-842). IEEE.

[35] Oliveira, V. A., & Conci, A. (2009). Skin Detection using HSV color space. In *H. Pedrini, & J. Marques de Carvalho, Workshops of Sibgrapi* (pp. 1-2).

[36] Soille, P. (2013). *Morphological image analysis: principles and applications*. Springer Science & Business Media.

[37] The pyOpenSSL developers. December 01, 2017. pyOpenSSL Documentation *Release 17.6.0.devo*.

[38] OpenSSL. Accedido en 2018.
https://www.openssl.org/

[39] Comandos útiles de OpenSSL. Accedido 2018.
https://soporte.itlinux.cl/hc/es/articles/202497479-Comando-%C3%BAtiles-de-OpenSSL

[40] Cómo filtrar el ruido de una máscara con openCV. Accedido en 2018.
https://robologs.net/2015/07/26/como-filtrar-el-ruido-de-una-mascara-con-opencv/

[41] Galería de fotos sobre psariasis | Psoriasis 360. Accedido en 2018.
https://www.psoriasis360.es/psoriasis/psoriasis-fotos